Interpretation of Association Rules with Multi-tier Granule Mining

JINGTONG WU

2014

Electrical Engineering and Computer Sciences

Science and Engineering Faculty

Queensland University of Technology

Submitted in fulfilment of the requirements for the degree of Doctor of Philosophy

Statement of Original Authorship

The work contained in this thesis has not been previously submitted to meet requirements for an award at this or any other higher education institution. To the best of my knowledge and belief, the thesis contains no material previously published or written by another person except where due reference in made.

Signature: QUT Verified Signature

Date: 08/May/2014

Acknowledgements

I would like to express my sincere gratitude to the people who have assisted and encouraged me during the research journey.

I wish to thank, first and foremost, my supervisor - Professor Yuefeng Li. Without his guidance and persistent help, this dissertation would not have been possible. Beside the research, Prof. Yuefeng Li is a well mentor demonstrates the positive attitude and the spirit of being diligent in my life.

I also wish to thank Professor Peter Bruza - my associate supervisor, Associate Professor Yue Xu and Dr. Yan shen for being the panel for my final seminar. They have given valuable advise that help me to complete my thesis.

Queensland University of Technology deserves my honest gratitude for offering an enjoyable working environment. Thanks to Science and Engineering Faculty, and my school, Electrical Engineering and Computer Science, especially. They have offered a number of training courses to improve my study and research capability. And thanks to the HDR student support team for their support greatly helped me to complete my study journey.

Special thanks to my previous colleagues, Dr. Daniel Tao, Dr. Yan shen and

Bin Liu. Their advices and shared experience are significant to solve the troubles during my study period.

This thesis is dedicated to my parents who have given me selfless love, early education and support throughout my life.

Many thanks to the thesis examiners, their valuable advices and constructive comments are important to refine the thesis to a better quality. Some of the general concerns have not been realized before the examination.

Finally, this thesis is proofread by Elite Editing. Thanks for their effort for making this thesis more readable.

Jingtong Wu

April 2014

Abstract

Association rule mining is an important data mining technology used to discover knowledge of items' association in data. It is usually a two-stage process: the first stage seeks to discover the frequent pattern, and the second stage seeks to generate association rules from the frequent patterns. It is currently a significant challenge to guarantee the quality of discovered knowledge in databases. The essential issue is to provide efficient methods to interpret meaningful discovered knowledge, such as methods to summarise and represent the discovered patterns and associations. One major obstacle to this is the overwhelmingly large volume of discovered patterns.

This research further develops the granule mining approach to improve the performance of association mining. Instead of patterns, this method uses granules to represent the knowledge implicitly contained in databases. This research also extends the approach to use a multi-tier structure and association mappings to summarise association rules in terms of granules. Consequently, association rules can be quickly accessed, and meaningless rules can be justified according to the association mappings. Moreover, the proposed structure is a precise compression

of patterns that can restore or estimate supports for given patterns.

Experiments are conducted to examine the performance of the proposed approach from several aspects, including the space and time complexity, scalability and accuracy of support estimation. The results show that the multi-tier granule mining approach is a promising approach to improve data mining performance.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This chapter first presents the background of this research, including a brief introduction of data warehouse, data mining and association rules. This chapter also presents the challenges and limitations of association rule mining, and some current approaches to overcome these challenges and limitations. It then introduces the rough set, decision rules and granules upon which this research was based, with tasks for using granules in the interpretation of association rules. Finally, it discusses the significance and contribution of this research.

1.1 Background

The volume of transaction data has become so large that there is a need for tools to perform analyses of these data. Data warehouses and data mining are such

tools, as technologies that make use of plain transaction records. Data warehouses are multi-dimensional databases that store data in data cubes. Each data cube contains a set of well-defined dimensions, with each dimension organised in hierarchies. A data cube also has measures which are aggregations based on the dimension hierarchies. Data warehouses provide tools to allow data to be presented from different aspects and abstract levels with structured information. With these features, data warehouses can be used as data sources for other analysis tools, such as data mining, to improve performance.

Data mining is the automatic acquisition of patterns that represent the knowledge implicitly stored in the data. Association rule mining is one form of data mining that discovers associations among attributes or items of transactions. There are two stages included in the association rule mining process: the pattern mining stage and the rule generation stage. The former is the process used to obtain frequent patterns, where a pattern is a set of items or attributes, and is considered a frequent pattern if it occurs in many transactions. The latter stage generates association rules from the discovered frequent patterns.

Traditionally, association rules only contain items from one dimension, while multi-dimension association rules have items from more than one dimension. Multi-dimensional association rules can be mined directly from data warehouses because these warehouses are multi-dimensional in natural. Further, some approaches are proposed to improve performance using the features and structural information provided by data warehouses, such as meta-rule methods. Finally,

based on the hierarchy information, new type of rules can be discovered, such as multi-level rules.

1.2 Problems of pattern mining based approaches

Efficient association rule mining algorithms have been developed to discover knowledge [3, 4, 27, 63, 71, 97] and approximation knowledge [1, 21, 61] in databases. However, there are some difficulties involved in applying these approaches to solve real problems [15, 50, 54, 103, 104] and in interpreting the discovered knowledge [48, 49, 59, 64]. These problems stem from the reasons listed below.

First, the algorithms generate an overwhelmingly large volume of patterns and rules, and the processes of pattern and rule generation are time consuming. This large volume makes it infeasible to use and maintain the discovered knowledge using traditional knowledge engineering techniques because of the combinational explosion in the number of discovered patterns.

Second, because the only relationship between the patterns is a subset("$\subset$") relationship, it is difficult to use the structural information provided by data warehouses to interpret item-based patterns in terms of data dimensions or categories. Moreover, the huge number of patterns causes an interpretability issue [64, 112]. Therefore, the unintelligibility of the discovered patterns and rules significantly obstructs their effective use; thus, the worth and importance of automatic

knowledge discovery depreciates significantly.

Third, there is noise and uncertainty contained in the discovered knowledge. The larger the volume of the results, the greater redundancy exists in the patterns and rules. Moreover, there are many unnecessary patterns and rules included in the results that are not interesting for users.

Finally, these patterns and rules are incomplete for knowledge coverage because the approach uses two approximation phases: minimum support for the pattern mining phase and minimum confidence for the rule generation phase. These two criteria miss some specific patterns or rules (such as the low-support problem, in which a large pattern is more specific, but with a very low support [45]). Lowering the support and confidence can improve the coverage; however, this generates a larger volume of results.

For all the above reasons, it is a significant challenge for association rule mining to effectively represent and interpret discovered patterns and rules. Thus, some approaches have been proposed to tackle these problems from different aspects. For frequent patterns, two forms of method can be used to reduce the number of patterns. The first is to produce fewer patterns during the pattern-generation process. Using a frequent closed pattern for frequent pattern mining is one such approach that has been proven by previous research [45, 103] to partially alleviate the redundancy problem. The second is the post-processing method, which summarises or compresses frequent patterns into a limited number of representatives. These approaches include pattern compression [106], pattern

deploying [103] and pattern summarisation [84, 112].

For rule generation, the main task is to eliminate redundancy and remove unnecessary rules. One means of achieving this is through using some form of constraints in the rule generation process. There are two categories for constraints: objective constraints and subjective constraints. For subjective constraints, approaches are proposed to allow users to provide rule templates or patterns based on structured information of data in order to discover only unifying rules [8, 26]. Objective constraints are used to construct concise representations of rules without applying user-dependant constraints [109, 123].

By observing these approaches, one important finding is that the use of closed patterns can greatly reduce the number of extracted rules [72, 96]. However, a considerable amount of redundancy remains [111]. Further, there are some other patterns, which are not really needed by the users, remain as well [54]. Therefore, the size of closed patterns must be further reduced. Based on closed pattern mining, the existing techniques can only provide limited means to eliminate redundant association rules [73, 109, 123], rather than eliminating association rules that are potentially meaningless. Constraint-based techniques also attempt to reduce search spaces for mining frequent patterns based on some sort of constraints, such as anti-monotonicity, monotonicity and succinctness [10, 39, 41, 70, 81–83]. Another category of solution is the summarisation approaches [33, 84, 98, 112]; however, these are loss methods that carry errors when restoring the support of original patterns from the compressed patterns.

Formally, this research aims to address the following limitations of current pattern-based association approaches. The first problem is the overwhelmingly large volume of discovered patterns and rules, which is the main obstacle to performance. Second, patterns do not carry the structural information of the data themselves; thus, they cannot use this information to improve the interpretation of association rules. Finally, using the two approximation measures of minimum support and confidence cause this approach to miss some specific patterns or rules (for example, the low-support problem [45]). Subsequently, support estimation through summarisation does not achieve a lossless result. These difficulties motivated this research to seek an appropriate alternative solution for association rule mining.

1.3 Pattern mining to granule mining

Based on the discussion in the previous section, the current approaches to solve the above mentioned issues either only partially address the issues or are post-processing methods that suffer from the disadvantages of the two-stage mining process. Therefore, this research uses an alternative association rule mining method that is not pattern based and can generate rules without the two-stage process. One such method involves using decision rules in decision tables [19, 49, 76, 78].

The concept of decision rules, derived from rough set theory, is well accepted in the rough set community. Rough set theory was developed to deal with vagueness

related to precise reasoning about approximations of vague concepts [9, 87]. A database table can be compressed as a decision table by using a "group by" operation to group attributes or items. A row in the decision table is called a 'granule' or 'classification rule', and consists of some relevant attributes. The rows in a decision table were also reviewed as decision rules by dividing the set of attributes into two groups: condition attributes and decision attributes [19,42,49, 76,120]. Decision rules were used for rule-based classification [88,90,102,116,125] and for the construction of decision trees and flow graphs [78, 115].

The advantage of using decision rules is that it can reduce the two phases of association mining into one process. However, three problems arise when trying to directly use decision rules for association rule mining. The first problem is that we do not understand the relationship between patterns and granules (or decision rules). The second is that decision rules can only represent a small proportion of associations in databases, and do not describe the associations between different granules. The third problem is that decision tables and flow graphs do not support a way to identify meaningless rules and access rules efficiently.

This research generalises the concept of patterns to granules for data mining. Granules are predicates that describe some common features of sets of objects (such as records or transactions) [76]. Granules comprise better properties than do patterns [48], and using granules for the interpretation of association rules has the following advantages over patterns. Given certain constraints on the attributes of a table, granules can be generated at different abstract levels efficient-

ly. Unlike patterns, which are parts of transactions (i.e., items in transactions), granules can describe common features of sets of transactions for selected sets of attributes. They are more general than patterns and contain more structural information.

By employing these advantages of granules, this research aims to extend the scope of association rule mining and presents a theory of granule mining to efficiently represent and interpret association rules. To achieve this, the following tasks were completed in this research. First, a set of concepts and definitions were needed to formally express the relationship between granules and patterns. Second, methods were required to determine interesting granules and the efficient process to find the corresponding associations between granules–that is, a subject-oriented and integrated method was required to present meaningful association rules. Third, it was necessary to determine how to represent meaningful association rules efficiently. Finally, it was necessary to design methods to estimate pattern supports through the granule mining approach.

1.4 Significance and contribution

Via completion of the tasks stated above, this research formed the following contributions:

- The proposed method has the ability to represent patterns in terms of granules, using formal definitions and concepts.

- This research presents multi-tier structures and association mappings to manage associations between granules. A multi-tier structure is a graph in which vertices are granules and edges are associations between granules.

- The attributes in a multi-tier structure can be tailored to incorporate hierarchical levels of data dimensions, such that the granules in the same tier have the same size, and those in different tiers have a different size.

- This research presents efficient algorithms for generating association mappings to illustrate the association between granules inter-tiers. In this way, meaningless association rules can be adjusted according to these association mappings.

- This research presents a method to use granules and association mappings to estimate supports for patterns.

This research is significant because granule mining has the following distinct features:

- It provides a new way to represent and organise the association rule in terms of granules and association mappings between granules.

- Using granules can explicitly describe selected dimensions based on user constraints, such that associations can be organised and represented at multiple levels (or different data tiers or granules)

- Granules and their associations explicitly carry the structure information of

data dimensions, which represents important progress towards the effective use of association rules.

- Granules indicate compressed representation, and the number of granules is small. This is critical to solve the overwhelming number of patterns, which is a crucial obstacle to the effective use of association rules.

- The completeness of granules enables an exact estimation of support for frequent patterns.

1.5 Publication

The publications for this research is listed as follow:

- Wanzhong Yang, Yuefeng Li, Jingtong Wu and Yue Xu. Granule mining oriented data warehouse model for representation of multi-dimensional association rules. *IJIIDS*, 2(1):125-145, 2008.

- Jingtong Wu and Yuefeng Li. Granule Oriented Data Warehouse Model. In *proceedings of the Fourth International Conference on Rough Sets and Knowledge Technology (RSKT 2009)*, pp. 255 - 263, Gold Coast, Australia.

- Yuefeng Li and Jingtong Wu. Summarization of Association Rules in Multi-tier Granule Mining. *The IEEE Intelligent Informatics Bulletin*, 13(1):21-29, 2012.

- Yuefeng Li and Jingtong Wu. Interpretation of Association Rules in Multi-Tier Structures. *International Journal of Approximate Reasoning*, Accepted on 19/Apr/2014

1.6 Thesis organization

The remaining sections of this thesis are organised as follows. Chapter 2 presents the literature review that outlines the works related to this research. This includes the introduction of the concepts of data warehouses, association rule mining and related theories, such as rough set concepts. This chapter also describes some related works, such as multi-level mining, meta-rule-based mining and improvement constraint-based mining approaches. Chapter 3 presents the framework of this study's approach, including the formal concepts and theories of the granule mining approach. It also discusses the concept of association mappings and the generation of association mappings. Chapter 4 presents the support estimation, including an introduction of the pattern-based method and the details of the granule mining based method. Chapter 5 discusses the implementation of the approach, including the construction of the multi-tier structure of granules, and the discovery of meaningful associations. The algorithms are also included in this chapter. Chapter 6 presents an evaluation of the research, including the experiment, results and an analysis of the experiment results. The final chapter provides a conclusion to the research and discusses possibilities for future research.

Chapter 2

Literature Review

This chapter reviews the related technology concepts and previous studies, as well as the basic theory upon which this research is based. The first three sections present a brief introduction of data warehouse, association rule mining and frequent pattern mining. The following section discusses some related previous studies that have sought to overcome the limitations of traditional pattern-based association rule mining. After these review, the next section introduces the rough set theory and some basic concepts of granule mining. The discussion of the difference between pattern and granule base approaches is then presented in the follow section.

2.1 Data warehouse

A data warehouse is an application that contains a collection of data that are subject-oriented, integrated, non-volatile and time-variant, supporting the management's decisions [30]. Data warehouses have some natural advantages if used as the source of data mining because the data in these warehouses are easily accessible, have consistent presentation, are adaptive and are resilient to change [37].

Data warehouses are constructed based on a multi-dimensional data model, in which the data are stored in the form of data cubes. A data cube consists of a set of dimensions and a set of measures of aggregated numeric values. Each measure of the data cube is stored in a table–called a 'fact table'–that contains the keys from the dimension tables.

Each dimension has a set of attributes among which relationships exist. This is called a 'concept hierarchy'. Figure 2.1 presents the Geo concept hierarchy of the Customer dimension. This concept hierarchy defines the mappings from the lower, specific level to the higher, general level of concepts. These levels are the basic unit of aggregation for the data cube. For the Geo hierarchy, there are four levels: Country, State, City and Name. Hierarchies such as the Geo hierarchy are in a 'total' order. Another type of hierarchy is in a 'partial' order, such as the Time concept shown in Figure 2.2 for the Time dimension.

The data cubes in the data warehouse are n-dimensional (n-D), where n is equal to or greater than two. Figure 2.3 shows a two-dimensional (2-D) table view of the Sales cube, according to the dimensions Time and Item. When n is three

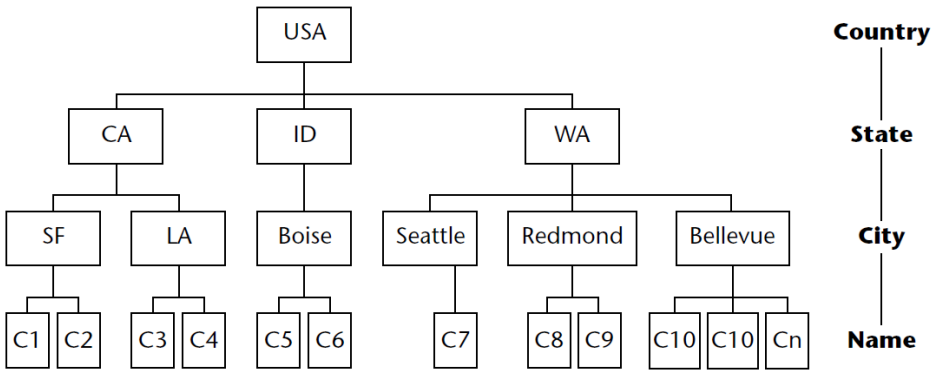


Figure 2.1: Customer dimension hierarchy [31]

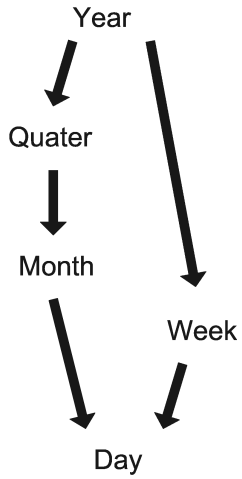


Figure 2.2: Partial order in time dimension

or greater, there are two types of views of n-D data. The first is a table consisting of a series of (n-1)-D tables, as shown in Figure 2.4. The second presents the data in a series of (n-1)-D cubes, as shown in Figure 2.5.

2.2 Data mining

The term 'data mining' is defined as extracting or 'mining' knowledge from a large quantity of data [25]. Data mining is also known as the synonym of the term 'knowledge discovery from data', or KDD. However, some view data min-

	Location="Brisbane"			
	Item (type)			
Time (quarter)	Home entertainment	computer	phone	security
Q1	605	825	14	400
Q2	680	952	31	512
Q3	812	1023	30	501
Q3	927	1038	38	580

Figure 2.3: 2D view of fact table

	Location="Brisbane"				Location="Sydney"				Location="Perth"			
	Item (type)				Item (type)				Item (type)			
Time (quarter)	home entertainment	computer	phone	security	home entertainment	computer	phone	security	home entertainment	computer	phone	security
Q1	605	825	14	400	605	825	14	400	605	825	14	400
Q2	680	952	31	512	680	952	31	512	680	952	31	512
Q3	812	1023	30	501	812	1023	30	501	812	1023	30	501
Q3	927	1038	38	580	927	1038	38	580	927	1038	38	580

Figure 2.4: 3D view of fact table

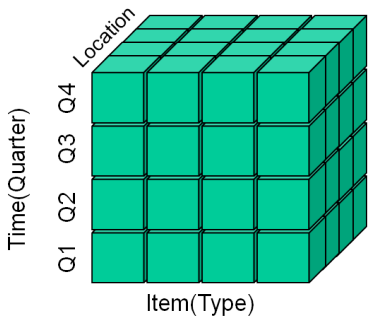


Figure 2.5: 3D data cube

ing as simply an essential step in the knowledge discovery process. The seven stages involved in the knowledge discovery process are illustrated in Figure 2.6. Specifically, steps one to four are the data preparation stage, during which variant data processing techniques are used. Some interaction with the user may also

occur here, so that the data are best tailored to fit the user's interest. Li and Ruan [43] proposed and extended process model which includes two additional steps. The first is the data collection step that collects data from real applications and previous mining results. The second is the processing step that falls between data selection and transformation. This step eliminates errors in the selected data. According to these steps, a typical data mining system consists of several corresponding components, as shown in Figure 2.7.

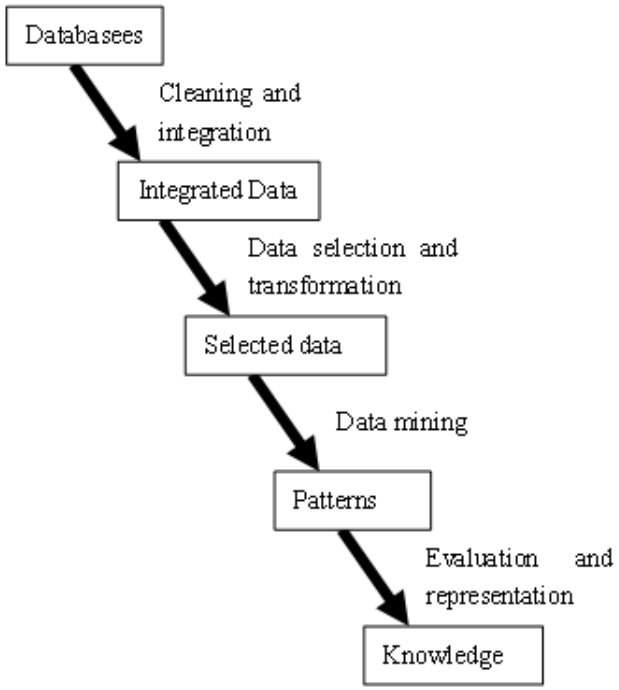


Figure 2.6: Data mining process

Currently, there are several different categories of data mining technology, based on the data mining task. These technologies include concept or class descriptions for associating data with concepts or classes, classification and prediction to find a model to distinguish between data classes, clustering the analyses

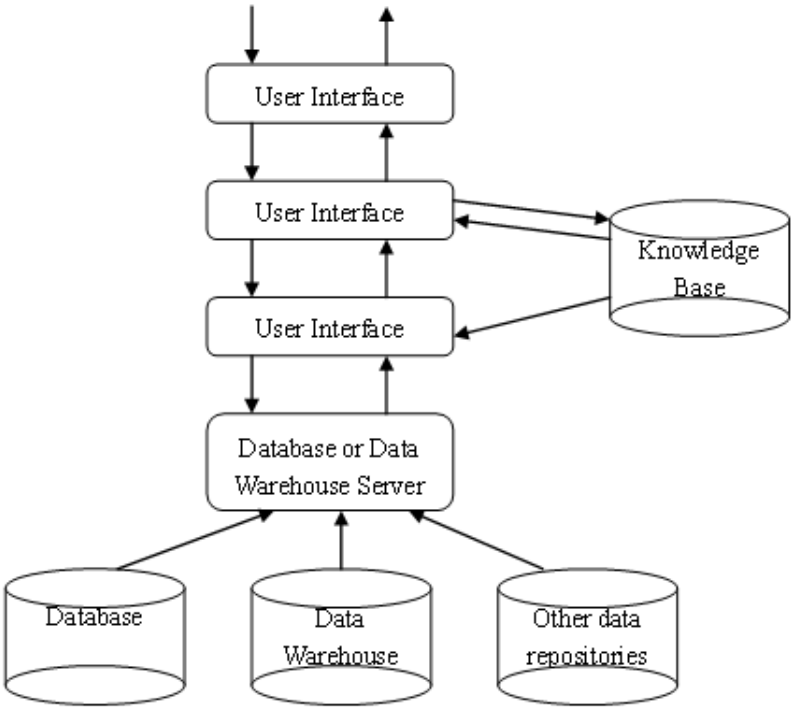


Figure 2.7: Components of data mining system

data without a known class label, and association mining that discovers correlations among data.

Association rule mining is the most important association mining technology to determine frequent patterns from transaction data, and generate association rules from the frequent patterns. Association rule mining has an effective application in the retail environment that is known as the 'shopping basket' analysis [66].

For example, association rule mining can be used to discover a customer's purchase patterns to help develop a better marketing strategy.

2.3 Association rule

Rule 2.1 is an example of an association rule indicating the knowledge that customers who buy bread, also buy milk. The support and confidence shown in the rule are the two measures of rule interestingness. Support of 10% means that 10% of customers buy bread and milk together, while a confidence of 80% indicates that 80% of the customers who buy bread also buy milk. The minimum support and minimum confidence can be used as the thresholds to determine frequent (or large) patterns and strong rules.

$$\text{Bread} \rightarrow \text{Milk} \quad [\text{support} = 10\%,\ \text{confidence} = 80\%] \tag{2.1}$$

Formally, the definition of support and confidence in association rule mining is as follows. Let $I = (i_1, i_2, ..., i_m)$ be a set of items. Let T be a set of transactions where each transaction, t, is a set of items, such that $t \subseteq I$. An association rule is an implication of the form $A \Rightarrow B$, where $A \subseteq I$, $B \subseteq I$, and $A \cap B = \emptyset$. The rule has support, s, in T if s percentage of the transactions in T contains $A \cup B$. Then s is the probability $P(A \cup B)$. The rule $A \Rightarrow B$ holds in the transaction set, T, with confidence, c, if c percentage of transactions containing A in T also contains B. Then c is the conditional probability $P(B|A)$. The equations to calculate support and confidence are as follows:

$$\text{support}(A \Rightarrow B) = P(A \cup B) \tag{2.2}$$

$$\text{confidence}(A \Rightarrow B) = P(B|A) = \frac{support(A \cup B)}{support(A)} \tag{2.3}$$

Traditionally, the target of mining association rules is to discover all association rules that have support and confidence greater than the minimum support and minimum confidence. Generally, there are two steps involved in the association rule mining process. The first finds all frequent patterns or itemsets that satisfy the minimum support. The second generates all association rules that satisfy the minimum confidence from the frequent patterns.

2.4 Frequent pattern mining algorithms

To efficiently acquire all the frequent patterns from the transactional database, different algorithms have been developed [2, 4, 27, 63, 71, 99]. Among these, there are two popular algorithms: the Apriori and FP-Tree algorithm.

The Apriori algorithm mines frequent patterns for Boolean association rules. It is an iterative process, known as a 'level wise search', in which the k-itemsets are used to generate (k+1)-itemsets. First, the frequent 1-itemset is generated by accumulating the count for each item in the database and collecting items with a larger support than minimum support. The result set is denoted L_1. L_1 then is used to find the frequent 2-itemset, L_2, which is used to generate L_3. The process continues until no frequent k-itemsets can be found.

The Apriori property–which requires that all non-empty subsets of a frequent itemset must also be frequent–is used to prune infrequent itemsets in the process

of generating frequent itemsets with two or more items. This process has two steps: the join step and the prune step. For the join step, a set of candidate k-itemsets, C_k, is generated by joining the itemsets in L_{k-1}. Let l_1 and l_2 be itemsets in L_{k-1}, such that $l_i[j]$ is the j-th item in l_i. Then the joint is performed, where members of L_{k-1} are joinable if they have (k-2) items in common. The second step starts when C_k is built. Each member in C_k is examined with the Apriori property. If a (k-1)-itemset is not frequent, it cannot be a subset of the frequent k-itemset, and the k-itemset of C_k with infrequent (k-1) subset is removed. Finally, the L_k set includes all the remaining itemsets in C_k after the pruning. The process ceases when no candidate itemsets can be found.

The FP-Tree algorithm was proposed by Han et al. [27]. It generates frequent itemsets without candidate sets in a tree-growing method. The generation of the FP-Tree starts with a root node labelled 'null'. The FP-Tree has a set of item prefix sub-trees as children, and a frequent-item header table. Each node in the sub-tree has three fields: item name, count and node-like. Each entry in the header table has two fields: item name and the head of the node-link.

The FP-Tree is constructed in two steps. In the first step, the database is scanned for a list of frequent items. In the second step, the root labelled 'null' is created, then the database is scanned a second time. According to the items in the transactions scanned, the new node is added to the tree, or the count of existing nodes in the corresponding path is increased by one for each occurrence.

After the FP-Tree is constructed, the frequent patterns can be mined from the

tree. The mining starts from the bottom of the header table. For one node, the conditional pattern tree is constructed first by scanning the FP-Tree for the paths to the node. Second, according to the conditional FP-Tree, the frequent pattern is discovered or the conditional FP-Tree is recursively mined. This process is completed when all members in the header table are used to discover the frequent patterns.

2.5 Multi-dimension association rule

Normally, when considering the dimension number of rules, association rules such as Rule 2.1 are called 'single-dimensional' or 'intra-dimensional' association rules because they only contain attributes from a single dimension. However, in a transactional database, the sales transactions always contain information of multiple dimensions, such as age, occupation and address. When treating each attribute as a predicate in the rule, one can mine the association rules consisting of multiple predicates, as shown in Rule 2.4. Association rules such as Rule 2.4 that contain two or more dimensions are multi-dimensional association rules.

$$\text{age}(X, 20|29) \wedge \text{occupation}(X, \text{student}) \Rightarrow \text{buys}(X, textlaptop) \tag{2.4}$$

For relational databases, there are three categories for mining multi-dimensional association rules, according to the manipulation method of the quantitative at-

tributes. The first is to mine the multi-dimensional association rules using static discretisation of quantitative attributes. The second dynamically discretises during the mining process [40, 91]. The third approach is the distance-based discretisation method that groups the attributes with close values into the same interval [62, 99].

Different to the relational databases, data warehouses store data in the multi-dimensional form naturally. Therefore, the mining can directly include the attributes from other dimensional tables as the predicates in the rule because the data cube or fact table has columns that link to the dimension tables.

2.6 Multi-level association rule mining

If considering the concept abstraction level, which is also a natural feature of data warehouses, it is possible to mine multi-level association rules. As shown in Figure 2.1, a concept hierarchy exists among the dimension attributes so that the association rules can be mined at each level, according to the hierarchy structure. Moreover, the association rules can also be mined between different levels of different dimensions. Finally, because the interestingness for the rules from different levels is different, flexible measurements and specific methods can be developed to mine patterns at multiple abstract levels and to generate the multi-level association rules.

Han et al. [24] proposed a method for mining multi-level association rules with an existing taxonomy of the database concept hierarchy. In this approach, the

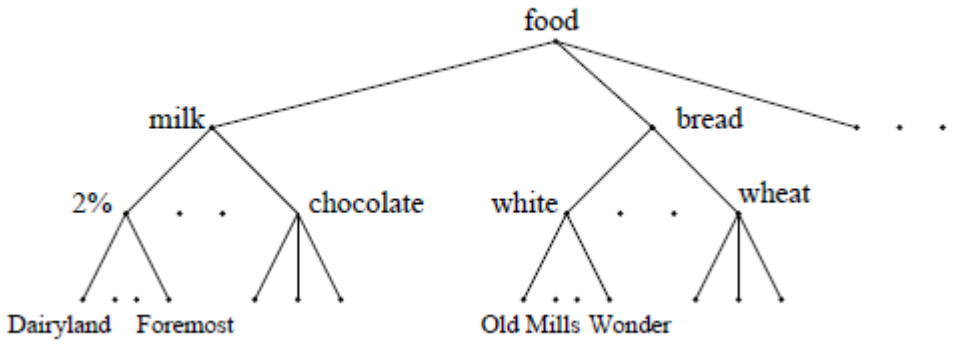


Figure 2.8: Multi-level hierarchy of product items [24]

definition of a strong multi-level rule is as follows. A pattern is frequent in set, S, at level l if its support is no less than the minimum support for the corresponding level. Then a rule '$A \rightarrow B/S$' is strong for a set, S, if it satisfies the following conditions. First, all ancestors (such as items in a higher level) of A and B must be frequent in their corresponding level. Second, the support of $A \wedge B$ must be frequent in the current level. Finally, the rule confidence must be no less than the threshold in the current level.

The mining method in this approach is a top-down method that starts from the highest level and ends at the lowest level, using the Apriori algorithm to mine frequent patterns and generate association rules.

Before the mining starts, the transactions are transformed into an encoded table as shown in Table 2.1, according to the concept hierarchy. In Table 2.1, every item of a transaction is turned into a digital code, where the number of digits is the same as the number of levels in the hierarchy, and each digit represents one concept level. In the sample table, the encoded item value is a three-digit

TID	Items
T_1	{111, 121, 211, 221}
T_2	{111, 211, 222, 323}
T_3	{111, 122, 211, 411}
T_4	{111, 121}
T_5	{111, 122, 211, 221, 413}
T_6	{211, 323, 524}
T_7	{323, 411, 524, 713}

Table 2.1: Encoded transaction table

number because there are three levels in the hierarchy, as shown in Figure 2.8. A variant method is also proposed to mine rules across different concept levels. The modification is that the discovery of frequent patterns uses not only the items from the current level, but also the generalised items from a higher level, except the items from the ancestors. For example, not only are items such as 11* and 12* used to find frequent patterns at the second level, but items such as 2** and 3** are used as well.

The mining process starts by scanning the whole table to discover the large 1-itemsets at level one. These itemsets are generalised items such as 1** and 2**. Following this, all frequent patterns are generated using the Apriori algorithm. Moreover, the items that are not large at this level, as well as the transactions that only have small items, are removed from Table 2.1 to generate the data for the next level of mining. For level two, the process is similar, and also starts from a large 1-itemset. The difference is that the items are generalised at level two, such as 11* and 12*. The entire mining process stops when it reaches the lowest level, or when no more candidate data remain.

This approach also provides methods to prune redundant and unnecessary

rules. A rule is redundant in terms of multi-level mining if it can be computed from a higher-level rule, and from the simple assumption of a relatively uniform distribution. To remove such rules, each strong rule at the current level is examined against all of its strong ancestor rules. If the confidence of this rule is not improved over a threshold, it is removed. For the unnecessary rule–that is not significantly different from a simpler rule, and is thus uninteresting–the pruning is as follows. For a strong rule, R: $A \rightarrow B$, it is examined against every rule R': $C \rightarrow B$, where $C \subseteq A$. If its confidence is not significantly different, this rule is excluded from the results.

2.7 Constraint based association rule mining

For the criteria for mining patterns and association, support and confidence are the two most basic measures. However, the mining process using only these criteria can produce a large number of patterns or rules that are unrelated or uninteresting to the users. One reason for this is because, although users know the best possible direction for mining, there is no facility for them to participate in the mining process. Therefore, constraint-based mining approaches are proposed to allow users to define the constraints for the rules to be mined. These constraints include knowledge type constraints, data constraints, dimensional and level constraints, interestingness constraints and rule constraints.

2.7.1 Meta-rule guided association rule mining

A meta-rule is one kind of constraint that is based on the analyst's experience, expectations or intuition regarding the data, or can be generated from the data schema. Meta-rules allow users to specify the syntactic form of the rules they expect. Meta-rule-guided mining is an interactive data mining method in which users propose the range of data for analysis by specifying hypotheses in the form of meta-rules or rule patterns.

Kamber et al. [35] defined the meta-rule as the template of a rule in the form:

$$M_R = P_1 \wedge P_2 \wedge ... \wedge P_m \Rightarrow Q_1 \wedge Q_2 \wedge ... \wedge Q_l$$

where P_i and Q_j are predicates or predicate variables, and m plus l is the number of predicates in the meta-rule. Rule R complies with M_R if, and only if, it unifies M_R.

In this approach, the mining is conducted on the multi-dimensional data cube $C[A_1,...,A_n]$, which is an n-D database. $A_1,...,A_n$ are n dimensions, and each dimension corresponds to a predicate in the meta-rule. Two methods–the multi-D-slicing and n-D cube searching method–were developed for mining rules complying with the meta-rule. The multi-D-slicing method first finds all large 1-predicate sets in p dimensions, then uses the large predicate sets in dimension (k - 1) to grow large k-predicate sets by multi-dimensionally slicing the data cube until all large predicate sets are discovered. The second method directly exam-

ines each p-D cell and adds the corresponding large p-predicate to the set of large predicates. The strong rules complying with the meta-rule are then generated from the large predicate sets.

In [16] and [65], two other methods are proposed to use a meta-rule and meta-pattern to perform association rule mining in data warehouse circumstances. These approaches also provide extra technologies, such as exploitation of the data hierarchy or new measures based on data warehouse features, to improve the rule interestingness.

Giuseppe and Pier [16] proposed a method to support the data mining process on data warehouses by exploiting the implicit and explicit concept hierarchies. This method explores the data schema and determines potentially interesting mining queries and a metric to estimate the rule interestingness.

The mining queries are also called 'mining patterns', and are used to define the regularity of the desired rules. In this approach, mining patterns are in the form of a tuple p: $\{T, g, m, s, c\}$, where T is the fact table, g is the grouping attributes, m is the rule attributes, and the remaining are the support and confidence. A simplified mining pattern is a mining pattern without the minimum support and confidence. The attributes in these patterns are either an attribute of T, or can be reached from an attribute through a dimension of T.

With the data schema of the data warehouse, the simplified mining patterns can be generalised by generalising the mining or rule attributes along the dimension hierarchy. After generalisation, the size of the rules and support are increased

because the mined values are decreased. This leads to overly large results and trivial rules. Therefore, a metric to identify the suitable generalisation is introduced. This metric is the average fraction of the distinct values of the attributes in each group, denoted as:

$$\tilde{f}_{g,m} = \frac{\tilde{a_{g,m}}}{|V_m|} = \frac{(\Sigma_{g_i} Z_i)}{(|V_g| \times |V_m|)}$$

where:

$$\tilde{a_{g,m}} = \frac{(\Sigma_{g_i} Z_i)}{|V_g|}$$

is the average number of distinct values of the rule attributes in a group, Z_i is the number of distinct values of m appearing in the i_{th} group g_i, $|V_g|$ and $|V_m|$ are the domains of g and m. The value of this metric increases when generalising a mining pattern, and reflects the increasing number of attribute values in a group. Thus, it can be used to decide whether the generalisation is suitable.

With a suitable metric, the proposed method can discover all potentially interesting meta-patterns, starting from the basic meta-patterns built for each attribute pair from T according to the schema. A lattice is built by generalising the basic simplified meta-patterns, as well as the generalised patterns. The generalisation is undertaken by grouping the mined and group attributes exploiting the dimension hierarchies. All generalised meta-patterns with less or equal value to the suitable metric are selected as the candidates for the user to perform data mining tasks.

Messaoud and Rabase'da [65] proposed a framework for mining inter-dimensional association rules from data cubes according to a sum-based aggregate measure. In this approach, the mining process is guided by a meta-rule context that is driven by analysis objectives. This approach also exploits aggregate measures to revise the definition of support and confidence.

The data cube used in this approach is depicted in Figure 2.9. A data cube, C, has a set of dimensions, including d dimensions, such that $D = \{D_i\}(1< i < d)$ and a set of measure M. Each $D_i \in D$ has a set of hierarchy levels. Then the $j_{th}(j \geqslant 0)$ hierarchical level of D_i is H_i^j. Let H_i be the set of hierarchy of dimension, D_i, then each level $H_i^j \in H_i$ consists of a set of member A_i^j. In Figure 2.9, a set of hierarchy in D_2 is $H_2 = \{H_2^0, H_2^1, H_2^2\} = \{\text{All, Family, Article}\}$, and a set of members of the Article level is $A_2^2 = \{\text{iTwin, iPower, DV-400, EN-700, aStar, aDream}\}$.

A sub-cube $D' \in D$ is a set of p dimension $\{D_1, ..., D_p\}$ from the cube C ($p \leqslant d$). The p-tuple $(\Theta_1, ..., \Theta_p)$ is called a 'sub-cube' on C according to $D' \iff \forall i \in \{1, ..., p\}$, $\Theta_i = \emptyset$ and there exists a unique j such that $\Theta_i \subseteq A_i^j$.

For a dimension, D_i, a dimension predicate, α_i, is in the form $\langle a \in A_i^j \rangle$. The dimension predicate takes a dimension member as a value. For instance, a dimension predicate in D_1 is: $\alpha_1 = \langle a \in A_{11} \rangle = \langle a \in \{\text{America, Europe, Asia}\} \rangle$.

Let $D' \in D$ be a set of p dimensions $\{D_1, \ldots, D_p\}$ from the data cube, C ($2 \leqslant p \leqslant d$). The predicate $(\alpha_1 \wedge ... \wedge \alpha_p)$ is called an 'inter-dimensional predicate' in $D' \iff \forall i \in \{1, ..., p\}$ where α_i is a dimension predicate in D_i. An inter-

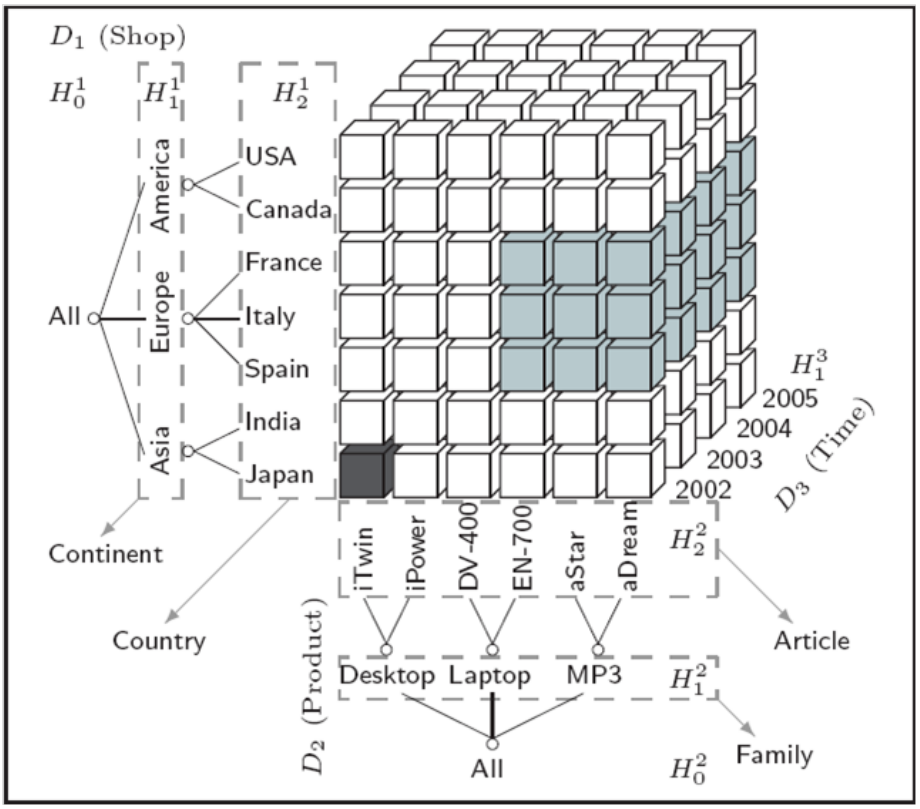


Figure 2.9: Example of the sales data cube [65]

dimensional predicate defines a conjunction of non-repetitive predicates. For example, for $D' = \{D_1, D_2\}$, $(\langle\alpha_1 \in A_{11}\rangle \wedge \langle\alpha_1 \in A_{11}\rangle)$ is an inter-dimensional predicate.

With these definitions, the inter-dimensional meta-rule can be defined. In the data cube, C, $D_C \subset D$ is a subset of p context dimensions. A sub-cube on C according to D_C defines the context of the mining process. D_A is a subset of analysis dimensions from which predicates of an inter-dimensional meta-rule are selected. Then an inter-dimensional meta-rule is of the following form: In the context $(\Theta_1, ..., \Theta_p)$, $(\alpha_1 \wedge ... \wedge \alpha_s) \Rightarrow (\beta_1 \wedge ... \wedge \beta_r)$, where $(\Theta_1, ..., \Theta_p)$ is sub-cube of C according to D_C. This defines a portion of the mining. The conjunction $(\alpha_1 \wedge ... \wedge \alpha_s) \Rightarrow (\beta_1 \wedge ... \wedge \beta_r)$ is an inter-dimensional predicate in D_A, where the number of predicates $(s + r)$ in the meta-rule is equal to the number of dimensions in D_A. An example of a meta-rule would be as follows:

In the context $(Student, Female), \langle a_1 \in Continent\rangle \wedge \langle a_3 \in Year\rangle \rightarrow \langle a_2 \in Article\rangle$

The actual mining is based on the Apriori algorithm, in which the items in the dimension predicates are specified by the sub-cube defined in the meta-rule. Another variation is that the support and confidence do not simply use the count, but the aggregated sum of counts of all the dimensions, for calculation. In the actual mining process, the strong rules are filtered by the modified support and confidence, and the two extra measurements are used to represent the interestingness of the mined rules.

2.7.2 Improvement constraint based association rule mining

Bayardo et al. [8] proposed another kind of constraint called 'improvement constraint'. This constraint examines the improvement of the confidence or predictive ability of a rule based on the sub-rule from which it is derived. For example, Rule 2.5 can be derived from Rule 2.6, and Rule 2.6 is a sub-rule or simplification of Rule 2.5. In this example, Rule 2.5 is considered an uninteresting rule because its confidence (95%) is 'significantly decreased' [8] from its sub-rule, Rule 2.6 (99%). A rule only can be considered interesting if its confidence is improved from its sub-rule.

$$Eggs\ \&\ Cereal \rightarrow Milk\ (Confidence = 95\%) \tag{2.5}$$

$$Cereal \rightarrow Milk\ (Confidence = 99\%) \tag{2.6}$$

The improvement constraint is defined as the minimum difference between its confidence and the confidence of any of its simplifications. This definition is formally defined as Rule 2.7, for a rule $A \rightarrow C$ and its simplified rule $A_s \rightarrow C$:

$$imp(A \rightarrow C) = MIN(\{conf(A \rightarrow C) - conf(A_s \rightarrow C) | A_s C A\}) \tag{2.7}$$

With this definition, the user can then specify the minimum improvement constraint to eliminate unnecessarily complex rules. The improvement constraint can be used in the mining process, with minimum support and confidence constraints, to mine only the rules whose confidence is at least greater than any of its simplifications for minimum improvement, where a simplification of a rule is formed by removing one or more conditions from its antecedent. Using the improvement constraint, many complex rules can be prevented from being generated unnecessarily. For example, as shown in Rule 2.8, for the simple rule $Cereal \rightarrow Milk$ with 99% confidence, there may be hundreds of complex rules that can be derived by adding conditions, such as I_1, I_2, ..., I_n with a confidence between 99% and 99.1%. All these rules can be avoided with the improvement constraint during the mining process.

$$Cereal\ \&\ I_1,\ \&\ I_2\ \&...\&\ I_n \rightarrow Milk \tag{2.8}$$

Moreover, the improvement constraint can also be used after the mining process to further remove uninteresting rules. These constraints can also be used to rank the mined rules according to each rule's improvement on confidence from its simplified rules in the results set.

This approach concludes that improvement constraints increase the efficiency of the mining algorithm and present users with a concise set of predictive rules that are easy to understand because every condition of each rule strongly contributes to its predictive ability.

2.8 Post-processing of frequent patterns

Using the features of the data warehouse to perform association mining can improve the efficiency and understanding of the discovered knowledge. However, the current approaches for association mining for data warehouses are mostly based on the traditional pattern-based mining methods. However, these methods still suffer from interpretation problems associated with the discovered patterns. Thus, some studies seek to overcome these difficulties by performing post-processing of the discovered patterns to provide more information about the patterns and reduce the volume of patterns.

Xin et al. [107] proposed a compress method that clusters the frequent patterns with a tightness measure, δ, and then selects the representative pattern from each cluster. First, a distance measure between two patterns is proposed to measure the similarity of the patterns. This distance is calculated through the coverage of each pattern. Second, to define quality-guaranteed clusters in order to minimise the number of representative patterns, a specific clustering criterion is used. In this criterion, a frequent pattern, P, is covered by the pattern, P', whose support is less than P within δ. This is also called 'δ cover'. The basic clustering is based on Greedy clustering and modified to cooperate with the FP-Tree algorithm. One method of clustering needs to compute the overall coverage for each pattern. Another method skips this, with some sacrifice of theoretical bound. A combination of these two methods is used to achieve a balanced performance.

Afrati et al. [1] proposed a method to approximate a collection of frequent

sets by using k sets that can succinctly represent the whole collection. These k sets are used to span an approximation of the original collection so that the intersection of these two collections is maximised. The spanning of k sets is the union of the power sets of each set in k. The result of spanning approximates the original collection within a factor of 1-$\frac{1}{e}$. There are two sources from which to select the spanner set. The first is the original collection. The spanner set selection is the '*MAX* k-COVER' problem. The second source is from outside of the collection. One source is the subset with a false positive ratio of the original collection. Another source is the border of the original collection, which causes some loss in approximation.

Yan et al. [112] proposed an approach to summarise the frequent patterns based on a profile solution that not only compresses the patterns, but also includes the support information, so that it can estimate the support for given patterns. The generation of the profiles is derived from the K clustering. Kullback-Leibler divergence of the distribution vectors is used to cluster the patterns into the profiles. When the pattern is inserted into the profile, the relative frequency of individual items in this profile is re-calculated. Moreover, each profile has a master pattern that contains all covered items, with its support calculated according to the cover set of the covered patterns. A profile is valid only when this support is frequent or δ-frequent; thus, the support of a given pattern covered by this profile can be calculated independently.

Wang et al. [98] proposed another summarisation method using probabilistic

models. The model used is the 'Markov Random Fields' (MRF) model, which exploits conditional independent relationships among the items in the transactional data. A k-itemset and its support represents a k-way statistic and can be a constraint on the distribution that generates the data. The process of summarisation is the construction of the MRF model. It starts from small itemsets and uses the model constructed by the small itemset to estimate the support of the larger itemset. If the itemset's support can be estimated with a δ error tolerance, then it is bypassed. Otherwise, it is used as an argument to generate the next level model. The process continues in iteration until all itemsets are processed. The resulting MRF model affords a concise and useful representation of the original collection of frequent patterns.

Jin et al. [33] proposed a regression-based approach for pattern summarisation, aiming to improve the quality of restoration. The optimisation of the restoration error is a non-linear least square optimisation problem; thus, this approach transforms the computation of the optimal parameter for restoration to a linear regression problem. Moreover, since multiple clusters can help improve the restoration, two new methods–K-regression and tree-regression clustering–are used to partition the frequent pattern collection to minimise the restoration error. The first method is derived from K-means clustering to obtain a minimum local restoration error, while the second method uses the decision-tree to minimise the total restoration error.

To precisely represent the entire collection of maximal frequent patterns, Jin

et al. [34] proposed a Cartesian contour representation based on the observation that shorter itemsets produce longer itemsets with the Cartesian product. In this approach, the frequent patterns are treated as vertexes of a bipartite graph, and the Cartesian product of these patterns is used as the edges in the graph. The union of these edges represents the entire collection. Thus, finding the concise representation of the entire collection is to seek a list of bi-cliques to cover the ground set with a minimal cost. This approach includes a general approach to complete this task. Several techniques are then developed to adapt this general approach to the Cartesian contour construction.

Poernomo et al. [84] proposed another profile-based summary method in order to solve the issues in pattern summarisation, including ease of analysing, good estimation and conciseness. In this approach, a special profile is developed to summarise the frequent patterns. This profile is called a 'CP-profile', and consists of a base set, a set of closures of the base set and the items covered. With this definition, the profiles are generated through finding the best items to create a profile that encompasses the maximum number of items with the least cost, to split the profiles' items into a base set and tail set. This setting also guarantees a ζ-adequate estimation.

Liu et al. [60] developed a method to find minimum representative patterns sets with less time consuming and memory usage. This method employs the greedy algorithm for the set cover problem to find representative patterns that can δ cover set of other patterns. For a given real number $\delta \in [0,1]$ and two

patterns X_1 and X_2, if $X_1 \subseteq X_2$ and their distance is less or equal to δ, then X_1 is δ-cover by X_2. Such that, for a frequent pattern set F with minimum support min_sup, there is a pattern set $\hat{F}$ with support no less than (1-δ)$\times min\_sup$. Then the minimum representative set is the minimum number of patterns in $\hat{F}$ that can δ-cover all patterns in F. Following techniques are used to improve efficiency. First, only closed patterns are considered. Second, CFP-tree structure–FP-tree for closed patterns–is used to find cover set efficiently. Final, the cover sets are compressed using a light-weight compressing technique.

2.9 Rough set and granule mining

This section introduces the rough set theory that this study bases on. It also introduces the basic concepts of granule mining. The rough set theory was introduced by Zdzislaw Pawlak. It is a mathematical tool used to deal with vagueness and uncertainty and an important theory for the artificial intelligence and cognitive sciences. While granule mining is developed based on rough set theory and uses the extended set to interpret association rules.

Rough set theory can be applied to many fields, including machine learning, knowledge acquisition, decision analysis, knowledge discovery from databases, expert systems and so forth. Particularly, there are some applications have been developed for network traffic data analyse using granule mining technology [57, 58].

2.9.1 Rough set

The rough set theory has some overlap with other methods handling vagueness and uncertainty, such as the Dempster-Shafter theory, which uses the belief function as its main tool, while the rough set theory uses sets of lower and upper approximations. One of the advantages provided by the rough set theory is that there is no need for any preliminary or additional information about the data. For example, the probability distribution in statistics and basic probability assignment in the Dempster-Shafer theory are not needed in the rough set approaches.

In the rough set theory, an information table is used to represent the input data of a domain in the real world, such as medicine, finance or transport. An example of the information table is shown in Table 2.2. Each row in the information table is called an 'example' ('objects' or 'entities'). Properties of examples are perceived by assigning values to some variables. There are two kinds of variables: 'attributes' (sometimes called 'condition attributes') and 'decisions' (sometimes called 'decision attributes'). Only a single decision is required in the information table. For example, in a hospital, patients are examples, while symptoms and tests are attributes, and diseases are decisions. Thus, each patient can be characterised by the results of tests and symptoms, and classified by disease.

The main concept of the rough set theory is indiscernibility, which is normally associated with a set of attributes. For example, let the set consist of headache and muscle pain in the information table shown in Table 2.2. Then e_1, e_2 and e_3 are characterised by having the same value of these attributes, so that e_1, e_2

	Attributes			Decisions
	Headache	Muscle_Pain	Temperature	Flu
e_1	yes	yes	normal	no
e_2	yes	yes	high	yes
e_3	yes	yes	very high	yes
e_4	no	yes	normal	no
e_5	no	no	high	no
e_6	no	yes	very high	yes

Table 2.2: An information table

and e_3 are indiscernible from each other. Hence, the indiscernibility relation is an equivalent relation. Sets that are indiscernible are called 'elementary sets'. In Table 2.2, the set of attributes of 'headache' and 'muscle pain' defines three elementary sets: $\{e_1, e_2, e_3\}$, $\{e_4, e_6\}$ and $\{e_5\}$. Moreover, any finite union of elementary sets is called a 'definable set'. For instance, $\{e_1, e_2, e_3, e_5\}$ is a definable set defined by the attributes 'headache' and 'muscle pain' by setting the values of both attributes to 'yes' or 'no'.

With the indiscernibility relation, redundant or dispensable attributes can be easily defined. If an attribute set and its super set define the same indiscernibility relation, then the attributes in the super set and those not in that set are redundant. For example, let 'headache' and 'temperature' be the set, and all three attributes be the super set. The elementary sets defined by the attribute set are all singletons–the same as the sets defined by the super set. Therefore, the attribute 'muscle pain' is redundant. Meanwhile, the attribute set Headache, Temperature has no redundant attribute because elementary sets of each attribute are not singletons. Such an attribute set is called 'minimal'.

Formally, a set, P, is the reduct of another set, Q, if P is minimal and both

	Attributes		Decisions
	Headache	Temperature	Flu
e_1	yes	normal	no
e_2	yes	high	yes
e_3	yes	very high	yes
e_4	no	normal	no
e_5	no	high	no
e_6	no	very high	yes

Table 2.3: Reduced information table

sets define the same indiscernibility relation. Set Headache, Temperature is a reduct of the original attribute set.

The elementary sets can also be defined by the decisions, and are called 'concepts'. As shown in Table 2.3, there are two concepts defined by the decision 'flu', which represents whether the patient has the flu or not. Decision 'flu' depends on the attributes 'headache' and 'temperature' because all elementary sets of indiscernibility relation associated with this attribute set are subsets of the concept 'flu' in Table 2.3. Thus, one can determine whether the patient is sick by the rules from the table shown in Table 2.3. For example, (Temperature, normal) $\rightarrow$ (Flu, no) or (Headache, no) and (Temperature, high) $\rightarrow$ (Flu, yes).

The data shown in Table 2.3 are consistent because all elementary sets belong to some concepts. However, the data in Table 2.4 are inconsistent because $\{e_5, e_7\}$ and $\{e_6, e_8\}$ are not subsets of any concepts. That means that decision 'flu' does not depend on the attributes 'temperature' and 'headache'. To solve this issue, the rough set theory introduces the lower approximation, which is the least definable set contained in each concept, X, and the upper approximation, which is the greatest definable set contained in X.

	Attributes		Decisions
	Headache	Temperature	Flu
e_1	yes	normal	no
e_2	yes	high	yes
e_3	yes	very high	yes
e_4	no	normal	no
e_5	no	high	no
e_6	no	very high	yes
e_7	no	high	yes
e_8	no	very high	no

Table 2.4: Inconsistent information table

In Table 2.4, for the concept $\{e_2, e_3, e_6, e_7\}$, the lower approximation is $\{e_2, e_3\}$ and the upper approximation is $\{e_2, e_3, e_5, e_6, e_7, e_8\}$, as shown in Figure 2.10. Sets such as $\{e_5, e_6, e_7, e_8\}$ that contain elements from the upper approximation that are not in the lower approximation are called a 'boundary region'. Thus, rough sets can be defined as sets that have non-empty boundary regions. With these tools, rules can be classified as follows: rules from the lower approximation are certainly valid, and rules from the upper approximation of the concepts are possibly valid.

The quality of the upper and lower approximation is used as the measure of uncertainty in the rough set theory. Given a set of examples, X, the quality of lower approximation is the ratio of the number of all elements in the lower approximation of X to the total number of X. It is the same for the upper approximation. For concept $X = \{e_1, e_4, e_5, e_8\}$ in Table 2.4, the quality of the upper approximation is 0.75(6 elements out of 8). The quality of both approximations can also be calculated as the ratio of a number of certain (for lower) or possible (for upper) examples to the total number of examples in X. Thus, the

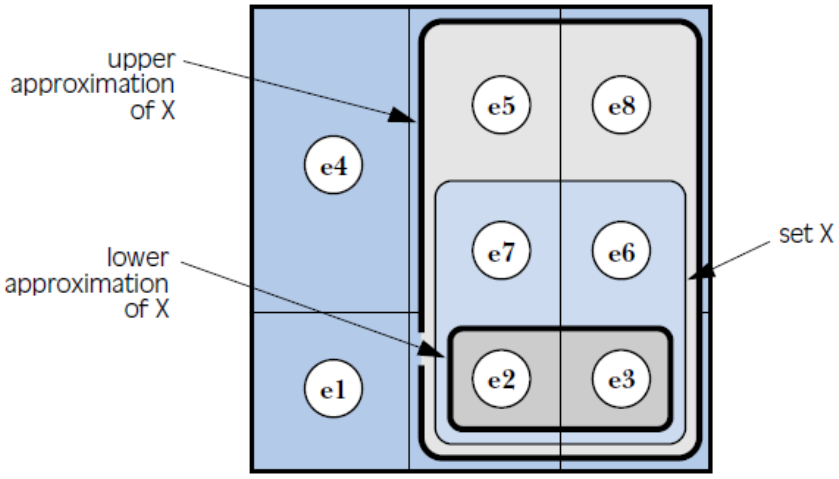


Figure 2.10: Lower and upper approximations [79]

quality is a relative frequency and this measure of quality is an objective belief function.

2.9.2 Granule mining

Granule mining is derived from the rough set theory–it is the essential technology upon which this research is based. Granule mining technology uses granular computing to interpret association rules through using extended sets that can be interpreted as a probability function or a belief function.

The first step of the granule mining approach is to transfer the database into a decision table, while the attributes selected to appear in the decision table must be divided into condition and decision groups. A decision table is defined as follows. The tuple (T, V^T, C, D) is used to denote the decision table of (T, V^T) such that the condition attribute, C, and decision attribute, D, satisfy $C \cap D = \emptyset$ and $C \cup D \subseteq V^T$. In dataset, T, for every attribute, $a \in V^T$, and the set of

Object	Items
t_1	a_1, a_5, a_6, a_7
t_2	a_1, a_5, a_6, a_7
t_3	a_1, a_2, a_4, a_5, a_7
t_4	a_1, a_2, a_3, a_5, a_6
t_5	a_1, a_2, a_4, a_5, a_6
t_6	a_1, a_2, a_3, a_5, a_6
t_7	a_1, a_4, a_5, a_7

Table 2.5: Transaction table

Granule	a_1	a_2	a_3	a_4	a_5
cg_1	1	0	0	0	1
cg_2	1	1	0	1	1
cg_3	1	1	1	0	1
cg_4	1	0	0	1	1

Table 2.6: Condition granules

all values of a, V_a, there is a mapping $a : T \rightarrow V_a$. For an object, $t \in T$, $a(t)$ denotes its value of attribute, a. For two objects, t_1 and t_2, if, and only if, $a(t_1) = a(t_2)$ for every $a \in C$, then $(t_1, t_2) \in I(C)$ is a binary relation on T determined by C. Thus, C defines a partition of all equivalent classes of $I(C)$ that is denoted as T/C or U/C. These classes are referred to as C-granules (or D-granules for T/D). The class that contains t is a C-granule induced by t, and is denoted by $C(t)$.

For example, for the data set shown in Table 2.5, let $T = \{t_1, t_2, ..., t_7\}$, $V^T = \{a_1, a_2, ..., a_7\}$. Tables 2.6 and 2.7 list the C-granules and D-granules where $C = \{a_1, a_2, a_3, a_4, a_5\}$ and $D = \{a_6, a_7\}$. With these definitions, we have $T/C = \{cg_1, cg_2, cg_3, cg_4\}$ and $T/D = \{dg_1, dg_2, dg_3\}$. A decision table can then be generated, as shown in Table 2.8, which contains five decision rules.

With the condition granules and decision granules, a mapping Γ from U/C to

Granule	a_6	a_7
dg_1	1	1
dg_2	0	1
dg_3	1	0

Table 2.7: Decision granules

Granule	a_1	a_2	a_3	a_4	a_5	a_6	a_7	Ng
g_1	1	0	0	0	1	1	1	2
g_2	1	1	0	1	1	0	1	1
g_3	1	1	0	1	1	1	0	1
g_4	1	1	1	0	1	1	0	2
g_5	1	0	0	1	1	0	1	1

Table 2.8: Decision table

$2^{(u/d)\times[0,1]}$ is determined by the set of decision rules, such that:

$$\sum_{(fst,snd)\in\Gamma(c_i)} snd = 1$$

for all $c_i \in U/C$, where $\Gamma(c_i)$ is the set of the decision granule-numeral pairs.

For each condition granule, its weight is the frequency in the decision table– that is, $w(c_i) = \sum_{x\in c_i} N_x$ where N_x is the number of transactions in class x. Normalising this weight attains a probability function, P, on U/C, such that:

$$P(c_i) = \frac{w(c_i)}{\sum_{c_j\in U/C} w(c_j)}$$

for all $c_i \in U/C$.

With Γ and P, an extended random set can be defined as a pair of (Γ, P). Then the decision rules can be represented as $c_i \to fst_{i,1}$, $c_i \to fst_{i,2}$, ..., and $c_i \to fst_{i,|\Gamma_{(c_i)}|}$ for a given decision granule, c_i, where $\Gamma(c_i) = \{(fst_{i,1}, snd_{i,1}), ...,$

U/C		$\Gamma(c_i)$	
c_1	$\rightarrow$	$(d_1, 0.8)$	$(d_2, 0.2)$
c_2	$\rightarrow$	$(d_2, 0.875)$	$(d_4, 0.125)$
c_3	$\rightarrow$	$(d_2, 1/6)$	$(d_4, 5/6)$
c_4	$\rightarrow$	$(d_3, 1.0)$	

Figure 2.11: An extended random set

$(fst_{i,|\Gamma_{(c_i)}|}, snd_{i,|\Gamma_{(c_i)}|})$. The strengths of these decision rules are then expressed as $P(c_i) \times snd_{i,1}$, ..., $P(c_i) \times snd_{i,|\Gamma_{(c_i)}|}$, respectively. The corresponding certainty factors, $snd_{i,1}$, ... $snd_{i,|\Gamma_{(c_i)}|}$, are computed as follows:

$$snd_{i,j} = \frac{|c_i \cap fst_{i,j}|}{|c_i|}.$$

Interesting rules can be determined by calculating the strengths and certainty factors when given a set of extended random set (Γ, P). For example, as shown in Figure 2.11, let U/C be a set of condition granules and $\Gamma(c_i)$ be the conclusion of premise c_i (i = 1, ..., $\mid U/C \mid$). A decision rule $c_i \rightarrow fst_{i,j}$ is an interesting rule if $pr(fst_{i,j})$ - $pr(fst_{i,j})$ is greater than the minimum threshold. The pr is the probability on (U/D), calculated as follows:

$$pr : (U/D) \rightarrow [0, 1]$$

such that:

$$pr(d) = \sum_{c_i \in (U/C), (d, snd) \in \Gamma(c_i)} P(c_i) \times snd.$$

Besides using the frequency criterion, the extended random sets can easily

use other measures to discover interesting rules. The first measure is the weight function for condition granules, which is used in the scenario of multiple data collection, such as mining in multiple databases. This new weight on U/C is calculated as:

$$w(c_i) = (\sum_{x \in c_i} N_x) \times log(M/n_i)$$

for all $c_i \in (U/C)$, where M is the total number of databases and n_i is the databases containing c_i.

Another measure is the uncertainties of decision granules, which is a relative stable measure. A random set (ξ, P) can be derived from the extended random set Γ, P: ξ: $U/C \rightarrow 2^{U/D}$ such that $\xi(c_i) = \{fst | (fst, snd) \in \Gamma(c_i)\}$ for all $c_i \in (U/C)$. The random set determines a Dempster-Shafer mass function, m_P, on U/D, such that:

$$m_P(X) = P(\{c_i | c_i \in (U/C), \xi(c_i) = X\})$$

for every $X \subseteq U/D$.

This mass function then decides a belief function and plausibility function as follows:

$$bel_m : 2^{U/D} \rightarrow [0,1],\ pl_m : 2^{U/D} \rightarrow [0,1]$$

and

$$bel_m(X) = \sum_{Y \subseteq X} m_P(Y), pl_m(X) = \sum_{Y \cap X \neq \emptyset} m_P(Y)$$

for every $X \subseteq U/D$.

Then the interval of $[bel_m, pl_m]$ can be used to examine the correctness of the subjective judgement for descriptions.

2.10 Difference between pattern and granule based approach

The most significant difference between the pattern-based association rule mining methods and this study's model is that pattern-based methods are a two-step process, while this model involves a one-step process. Granules and basic mappings can be generated in one step; then, from these granules and mappings, granules of different size and derived mappings can be generated.

The second difference relates to the use of the structural information of data. Pattern-based models require extra effort to use structural information to improve mining performance. For example, meta-rules are created using the dimensional information of data cubes to define the mining space and rule template to mine. In contrast, due to the properties of granules, granules contain structural information naturally. For example, a granule can be defined by high profit products and low profit products.

Another difference is the ways they describe association. For association rules, the association only describes the subset relationship between a pattern and its subsets. In contrast, the granule-based model uses association mapping to de-

scribe the associations between granules of different sizes. Moreover, associations can be interpreted using structural information provided by data warehouses because granules carry such information naturally as a result of their properties.

2.11 Summary

This chapter has presented an introduction of data warehouses, which are multidimensional databases. These databases contain a set of data cubes that consist of a set of dimensions and the measures stored in the fact table. Data warehouses enable the inspection of data from different views and abstract levels, according to the data schema and hierarchical structure of the data.

This chapter also introduced association rule mining, which is a data mining technology used to discover the knowledge about the patterns and associations between the items of the transactions in the database. Association rule mining is a two-stage process: the frequent pattern discovery stage and the rule generation stage. Two algorithms for frequent pattern mining–the Apriori and FP-Tree algorithm–were illustrated in this chapter. These algorithms mine frequent patterns only, based on the occurring frequency, and do not use the features provided by the source database, such as data warehouses, to improve mining performance in terms of efficiency and effectiveness.

This chapter also reviewed some approaches [16, 35, 65] that have used association rule mining for data warehouses using the features of the data warehouse. In these approaches, meta-rules or meta-patterns were used as a template for the

desired rules. The meta-rules were designed using the dimension and hierarchy levels of the data warehouse. These approaches can limit the mining space in order to improve the mining efficiency. They also can increase the rule interestingness because the user can provide a template of desired rules.

In addition to the meta-rule-based approaches, this chapter also introduced another approach that uses a different type of constraint–the improvement constraint. This approach only mines the rules that have greater improvement in confidence than their simplified rules, where the simplified rules are generated by removing one or more attributes from the original rule. Using the improvement constraint, many unnecessary complex rules can be avoided.

Usually, the volume of the patterns and rules is overwhelmingly large, which leads to the interpretation issue. In response to this, some post-processing approaches have been proposed. This chapter reviewed some of these approaches [1, 33, 60, 84, 98, 107, 112] that summarise or compress the frequent patterns into a limited number of representatives. Moreover, methods were also proposed to restore the original supports for the patterns from the representatives.

Finally, this chapter introduced the rough set theory and granule mining, which is based on the rough set theory. The rough set theory and granule mining use a decision table, instead of traditional transactional data. Each row in the decision table is called a 'decision rule'. Thus, the pattern can be characterised by the condition and decision granules in order to interpret the association rules. Moreover, the relationships between the condition and decision granules gener-

ate a mapping–called an 'extended random set'–hat can be used to determine interesting rules via some measures. Thus, it is possible to use granule mining to overcome the limitations of patterned-based association mining, such as the time required for both pattern mining and rule generation, and the interpretation issue caused by the lack of semantic meaning for patterns. Therefore, this study's approach is based on the granule mining technology and is offered as a potential solution to resolve these issues.

Chapter 3

Framework

This chapter first presents an overview of this study's approach and the system architecture for implementation. Second, it discusses the differences between pattern mining and granule mining. Third, it describes the advantages and disadvantages of using granules in knowledge discovery, which leads to a discussion of the motivation behind this study's approach. Fourth, it illustrates the interpretation of decision rules in terms of association rules. Finally, it presents a multi-tier structure, as well as the details for deriving varieties of association mappings for the construction of the multi-tier structure.

3.1 Overview and system architecture

From the overview, the proposed approach performs the granule mining, including generation of granules, construction of the multi-tier structure and mappings of the data from the data warehouse. It firstly retrieves the data from the data warehouse, and then builds the information table and decision table according to the user's request and granule definitions. It then generates the granules and constructs a two-tier structure. Following this, it derives variates of association mappings from the two-tier structure and constructs the multi-tier structures. Finally, it conducts the mining tasks, such as meaningless rule filtering and support estimation, and presents the final results to the user.

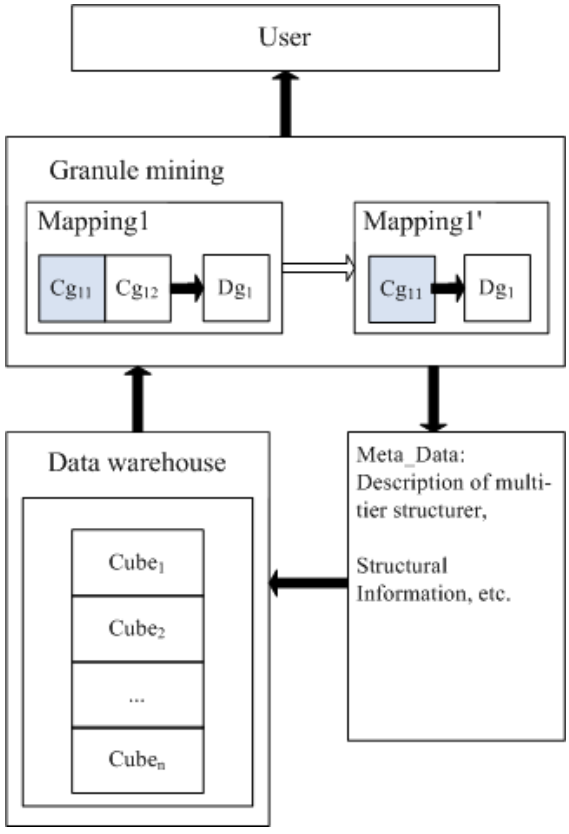


Figure 3.1: System architecture

For the system to implement the approach, the system architecture is designed as shown in Figure 3.1. There are corresponding components for each step of the process. The data warehouse component provides tools to use the data warehouse features to facilitate the mining process. For example, with the multi-dimensional data schema, it can easily involve the user in the mining process to provide constrains to define different granules. Moreover, the query language of the data warehouse enables a one-step retrieval of the data with specified attributes, and simplifies the data preparation and information table construction. Finally, the granule mining component is responsible for the tasks, including accepting the user constraints, constructing the multi-tier structure and mapping according to these requirements. It is also responsible for displaying the data mining results to the user after meaningless rule removal or support estimation.

3.2 Using data warehousing in our model

This study's model uses the features provided by data warehouses in two ways. First, it uses the structural information provided by the data warehouse to define granules, so that the structure of the granules carries this information naturally, and can be used to express association meanings. In experiments on the Foodmart 2005 dataset, the attributes of the 'Product' dimension were used to define the granules. These granules were defined at different abstract levels, according to the hierarchy levels of the 'Product' dimension. For example, attributes from the 'Product Family' can be used to define large granules, so that the associations of

the granules describe the relationships between product families. While 'Product Department' attributes can be used to define smaller granules, associations are interpreted as relationships between the product departments. Therefore, associations between granules of different sizes can be interpreted with the structural information contained in data warehouses.

Second, using tools for data warehousing, we can build an information table that meets a mining request simply in a one-step query. In our implementation, we used the SQL Server 2008 as the data warehousing platform. It provides a query language, DMX, to retrieve date. The data retrieved through this query language are in an integrated form. Therefore, we can retrieve data that match the data mining request, and transform the retrieved data into information table in straightforward steps through this query language provided by the data warehouse.

3.3 Data mining to granule mining

After a short preamble, and before turning to the theoretical side of granule mining in the following sections, it is worthwhile discussing the two perspectives of knowledge in data. This section discusses association rules and patterns by using an example that is also used to describe how decision rules and granules can be used. This section also discusses the differences between decision rules (granules) and association rules (patterns) that led to the motivations of developing granule mining.

Object (Transaction)	Items (Products)
t_1	a_1 a_2
t_2	a_3 a_4 a_6
t_3	a_3 a_4 a_5 a_6
t_4	a_3 a_4 a_5 a_6
t_5	a_1 a_2 a_6 a_7
t_6	a_1 a_2 a_6 a_7

Table 3.1: A transaction table

Object(Transaction)	a_1	a_2	a_3	a_4	a_5	a_6	a_7
t_1	1	1	0	0	0	0	0
t_2	0	0	1	1	0	1	0
t_3	0	0	1	1	1	1	0
t_4	0	0	1	1	1	1	0
t_5	1	1	0	0	0	1	1
t_6	1	1	0	0	0	1	1

Table 3.2: An information table

3.3.1 Association rules

Formally, a transaction database is a set of shopping lists that can be described as an information table based on some constraints. For example, a meta-rule (a predicate) can be used to select a set of items (products) in a period that satisfies certain conditions.

Let T be a set of objects (record) in which each record (object) is a sequence of items, and let $V^T = \{a_1, a_2, \ldots, a_n\}$ be a set of selected items (or attributes) for all objects in T, where each item can be a tuple (e.g. $< name, cost, price >$ is a product item). We call (T, V^T) an information table. Table 3.2 illustrates an information table, where $V^T = \{a_1, a_2, \ldots, a_7\}$, and $T = \{t_1, t_2, \ldots, t_6\}$.

A set of items is called an 'itemset' or 'pattern'. Given a pattern (an itemset), X, its support, *support(X)*, is the fraction of the objects (transactions) that

contain X. A pattern is considered a frequent pattern if its support is greater than or equal to a minimum support, min_sup.

Table 3.1 has 10 frequent patterns if we assume $min\_sup = 50\%$. These are

$\{a_1\}, \{a_2\}, \{a_3\}, \{a_4\}, \{a_6\}$ — 1-*itemset*

$\{a_1, a_2\}, \{a_3, a_4\}, \{a_3, a_6\}, \{a_4, a_6\}$ — 2-*itemset*

$\{a_3, a_4, a_6\}$ — 3-*itemset*.

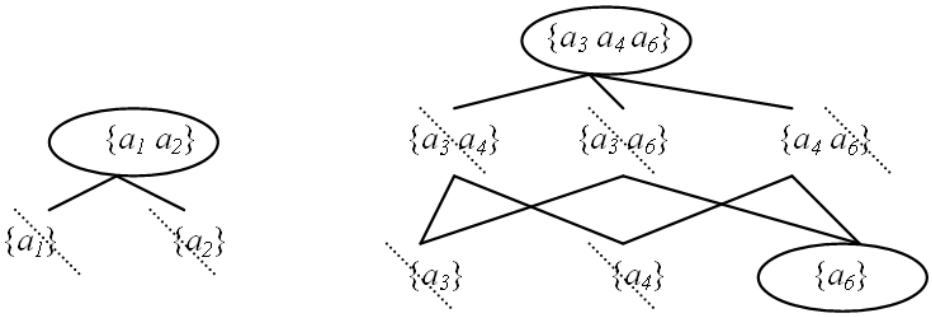


Figure 3.2: Pattern taxonomy

Not all frequent patterns are useful. For example, pattern $\{a_3, a_4\}$ in Table 3.1 is a non-closed pattern because it always occurs with item a_6 in all transactions– that is, $support(\{a_3, a_4\}) = support(\{a_3, a_4, a_6\})$. There are only three closed patterns in the above example: $\{a_3, a_4, a_6\}$, $\{a_1, a_2\}$, and $\{a_6\}$.

Patterns can be discussed in a taxonomy by using the subset relation. Figure 3.2 illustrates an example of the pattern taxonomy for the frequent patterns in Table 3.1, where the nodes represent frequent patterns, and the edges are 'subset' relations. We can prune non-closed patterns in the pattern taxonomy (see the nodes crossed by a dashed line). In that case, some direct 'subset' relations

may be redirected. For example, pattern $\{a_6\}$ would become a direct sub-pattern of $\{a_3, a_4, a_6\}$ in Figure 3.2 after pruning all non-closed patterns.

We can generate association rules from these closed patterns. Let A be a closed pattern, and $B \subset A$. Rule '$B \rightarrow (A - B)$' is called an 'association rule', and its confidence is the percentage of transactions containing B that also contain $(A-B)$–that is the conditional probability $P((A-B)|B)$. We call '$B \rightarrow (A-B)$' an interesting rule if $P((A-B)|B) > P(A-B)$, where $P(A-B) = support(A-B)$, and $P((A-B)|B) = \frac{support(A)}{support(B)}$. In real applications, people usually use a minimum confidence, *min_conf*, to determine interesting rules.

For example, from closed pattern $\{a_3, a_4, a_6\}$; we have the following three association rules with the longest antecedents and 100% confidence since $support(\{a_3, a_4, a_6\}) = support(\{a_3, a_6\}) = support(\{a_4, a_6\}) = support(\{a_3, a_4\})$:

$$a_3 \wedge a_6 \rightarrow a_4;\ a_4 \wedge a_6 \rightarrow a_3;\ a_3 \wedge a_4 \rightarrow a_6. \tag{3.1}$$

3.3.2 Decision rules

The information table can also be represented in granules according to user constraints. A granule is a predicate that describes the common features of a set of objects (such as records or transactions) for a selected set of attributes (or items). Granules can be used to tailor multi-dimensional databases in order to meet user constraints for knowledge discovery in databases.

The simplest case of using granules is to group items (such as products) into

Granule	Desktop Family							N_g	coverset
	High Profit Products					Low Profit Products			
	a_1	a_2	a_3	a_4	a_5	a_6	a_7		
g_1	1	1	0	0	0	0	0	1	$\{t_1\}$
g_2	0	0	1	1	0	1	0	1	$\{t_2\}$
g_3	0	0	1	1	1	1	0	2	$\{t_3, t_4\}$
g_4	1	1	0	0	0	1	1	2	$\{t_5, t_6\}$

Table 3.3: A decision table

two categories: condition attributes and decision attributes. For example, users may view high profit products as condition contributes and low profit products as decision attributes. Let a_1, a_2, a_3, a_4 and a_5 be the high profit products (or condition attributes) that are used to form the antecedents of rules, and a_6 and a_7 be the low profit products (or decision attributes) that are used to form the consequents of rules.

Table 3.3 illustrates a decision table of Table 3.1, where the semantic information is used to group items (products) into categories: *Desktop*, *Family*, *High Profit* and *Low Profit*. The set of granules is $\{g_1, g_2, g_3, g_4\}$, where N_g is the number of objects that have the same representation (granule), and the coverset is the set of objects (transactions) that are used to produce a granule.

Every granule in the decision table can be mapped into a decision rule [77], where we treat the presence and absence of items as the same position if we view the decision table as a database. Therefore, we can obtain four decision rules from Table 3.3, and the second granule, g_2, can be read as the following decision

Condition Granule	a_1	a_2	a_3	a_4	a_5	coverset
cg_1	1	1	0	0	0	$\{t_1, t_5, t_6\}$
cg_2	0	0	1	1	0	$\{t_2\}$
cg_3	0	0	1	1	1	$\{t_3, t_4\}$

Table 3.4: C-granules

Decision Granule	a_6	a_7	coverset
dg_1	0	0	$\{t_1\}$
dg_2	1	0	$\{t_2, t_3, t_4\}$
dg_3	1	1	$\{t_5, t_6\}$

Table 3.5: D-granules

rule:

$$(a_1 = 0 \wedge a_2 = 0 \wedge a_3 = 1 \wedge a_4 = 1 \wedge a_5 = 0) \rightarrow (a_6 = 1 \wedge a_7 = 0), \tag{3.2}$$

where the antecedent and consequent are described as Boolean expressions.

The smallest granules only contain one single attribute–these are also called 'primary granules'. A large granule can be generated from some smaller granules by using logic 'and', $\wedge$. For example, granules can be defined based on condition attributes and decision attributes. The former is called 'C-granule' (condition granule) and the latter is D-granule (decision granule). Tables 3.4 and 3.5 show the C-granules and D-granules, respectively. The large granules g_1, g_2, g_3 and g_4 can be generated by C-granules and D-granules as follows:

$$g_1 = cg_1 \wedge dg_1;$$

$$g_2 = cg_2 \wedge dg_2;$$

$$g_3 = cg_3 \wedge dg_2;$$

$$g_4 = cg_1 \wedge dg_3.$$

Figure 3.3 illustrates a two-tier structure [49] to describe the relationship between these granules in Tables 3.3, 3.4 and 3.5. The links (arrows) also represent the associations (decision rules) between condition granules and decision granules.

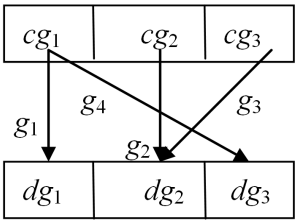


Figure 3.3: A 2-tier structure

Notice that the decision rules, as shown in Eq. 3.2, can now be simply described as follows:

$$cg_2 \rightarrow dg_2 \quad (g_2) \tag{3.3}$$

where the rule is corresponding to a granule (g_2), and the antecedent and consequent are a C-granule and a D-granule, respectively. This rule shows the association between high profit products and low profit products, which includes much more semantic meaning than the rules in Eq. 3.1, which only shows associations between items. In addition, only one rule $a_3 \wedge a_4 \rightarrow a_6$ in Eq. 3.1 is useful based on the above user constraints; however, it is impossible to recognize it before the phase of the rule generation.

3.4 Motivation

As previously indicated, association rule mining includes two phases: pattern mining and rule generation. The first phase of pattern mining is the discovery of frequent patterns. The second phase of rule generation is the discovery of the interesting and useful association rules in the discovered patterns. The first phase often takes a long time to discover all frequent patterns, and the results tend to include a lot of noise. The second phase is also time consuming and can generate many redundant rules. It is worth pointing out the issues associated with the two phases in association mining, which are that both phases take a long time and contain many uncertain actions for finding useful knowledge in databases. In terms of granules, it is clear that knowledge in databases can be discovered in a single phase–granule mining–because the discovered rules are indeed granules. Based on the discussion in the previous section, the following advantages of using granules have also been identified:

1. A decision table can directly describe multiple values of items, and provide a user-oriented approach to determine the antecedent (premise) and consequence (conclusion) of association rules in databases.

2. A granule is a predicate that describes the features of a set of objects in decision tables.

3. Granules can be divided into smaller granules based on categories of attributes, and smaller granules can be merged into larger granules.

4. The number of granules is much smaller than the numbers of patterns in databases.

Certainly, there are also several disadvantages when discussing granules based on decision tables only. It should be obvious at this point that we do not understand the relationship between association rules (or patterns) and decision rules (or granules), although they appear very similar in the previous example. Decision tables can provide an efficient way to represent discovered knowledge with a small number of attributes; however, in cases of large numbers of attributes, decision tables lose their advantages because they do not provide a mechanism for dealing with granules of different sizes. In addition, decision tables cannot be used to discuss meaningless rules because they do not provide a semantic structure for discussing generalised rules.

3.5 Interpreting the granules in terms of patterns

In order to represent the knowledge discovered in the database through the granules, the decision rules need to be interpreted in terms of association rules. This interpretation is achieved by exploring the relationship between the granules and patterns–especially the closed patterns. The exploration begins by introducing the definitions of *coverset* of patterns, *closure* and *coverset* of granules in the granule mining approach, and then using these concepts to define the relation-

ship between patterns and granules, as well as some theorems deduced from these definitions.

The transaction database that is used to obtain patterns or build decision tables can be formally denoted as a simple information table $(T,\ V^T)$, where T is the set of transactions in which every transaction contains a sequence of items, and $V^T = \{a_1,\ a_2,\ ...,\ a_n\}$ is the set of items of patterns or attributes of granules for all transactions in T. With the denotion of the information table, the definition–including coverset, closure and closed patterns–can be introduced first in regard to patterns.

Definition 1: *Coverset* of an itemset. For an *itemset*, X, that is a subset of V^T, the *coverset* of X is defined as the set of all transactions or objects $t \in T$ such that $X \subseteq t$. That is, $coverset(X) = \{t | t \in T,\ X \subseteq t\}$. Because the occurrence frequency of an *itemset* X is the number of transactions containing X which is equal to $|coverset(X)|$; thus, the support of X is $\frac{|coverset(X)|}{|T|}$. An *itemset* X is considered a frequent pattern if its support is greater than minimum support which means the *coverset* of X is larger than the minimum threshold.

Definition 2: *Itemset* of a set of transactions. The set of items or attributes that appear in all the transactions of a set of transactions, Y, is the *itemset* of Y. It is denoted as follows:

$$itemset(Y) = \{a | a \in V^T, \forall t \in Y \Rightarrow a \in t\}.$$

Definition 3: *Closure* of an *itemset*. For a pattern, X, its closure $closure(X)$

Frequent Pattern	coverset	closure	closed?	max?
$\{a_1\}$	$\{t_1, t_5, t_6\}$	$\{a_1, a_2\}$		
$\{a_2\}$	$\{t_1, t_5, t_6\}$	$\{a_1, a_2\}$		
$\{a_3\}$	$\{t_2, t_3, t_4\}$	$\{a_3, a_4, a_6\}$		
$\{a_4\}$	$\{t_2, t_3, t_4\}$	$\{a_3, a_4, a_6\}$		
$\{a_6\}$	$\{t_2, t_3, t_4, t_5, t_6\}$	$\{a_6\}$	*yes*	
$\{a_1, a_2\}$	$\{t_1, t_5, t_6\}$	$\{a_1, a_2\}$	*yes*	*yes*
$\{a_3, a_4\}$	$\{t_2, t_3, t_4\}$	$\{a_3, a_4, a_6\}$		
$\{a_3, a_6\}$	$\{t_2, t_3, t_4\}$	$\{a_3, a_4, a_6\}$		
$\{a_4, a_6\}$	$\{t_2, t_3, t_4\}$	$\{a_3, a_4, a_6\}$		
$\{a_3, a_4, a_6\}$	$\{t_2, t_3, t_4\}$	$\{a_3, a_4, a_6\}$	*yes*	*yes*

Table 3.6: Closed patterns

$= itemset(coverset(X))$. This definition originally comes from the Galois connection [14]. There are some important properties of this definition as follows. It is easy to prove these properties (see [14], [123] or [109]).

Theorem 1. Let X and Y be two patterns, we have

(1) $X \subseteq Y \Rightarrow coverset(X) \supseteq coverset(Y)$;

(2) $coverset(X \cup Y) = coverset(X) \cap coverset(Y)$;

(3) $coverset(X \cap Y) \supseteq coverset(X) \cup coverset(Y)$;

(4) $closure(X) \supseteq X$ for all patterns X;

(5) $X \subseteq Y \Rightarrow closure(X) \subseteq closure(Y)$ for all patterns X and Y.

Definition 4: Closed pattern. A pattern, X, is called closed if, and only if, $X = Closure(X)$. X is considered a 'max closed pattern' if all its super patterns are non-closed, where pattern Y is called a 'super pattern' of X if $Y \supset X$.

Using the above definitions, the frequent patterns can be described with more information. Table 3.6 lists the additional properties of each frequent pattern obtained from Table 3.1 using the new definitions.

Regarding the definition of granules, we need to introduce the denotion of the decision table and the equivalence class for granules. For an information table (T, V^T), the tuple (T, V^T, C, D) is used to denote the decision table generated from (T, V^T) if $C \cap D = \emptyset$ and $C \cup D \subseteq V^T$. With the assumption that there is a function for every attribute $a \in V^T$ such that a: $T \to V_a$, where V_a is the set of all values of a. V_a is also called the domain of a. For example, $V_a = \{1, 0\}$ in Table 3.3.

Given a subset of V^T, B, it can determine a binary relation $I(B)$ on T such that $(t_1, t_2) \in I(B)$ if, and only if, $a(t_1) = a(t_2)$ for every $a \in B$, where $a(t)$ denotes the value of attribute, a, in the transaction, $t \in T$. It can be proven that $I(B)$ is an equivalent relationship. The family of all equivalence class of $I(B)$, which is a partition determined by B, is denoted as T/B.

For the decision table (T, V^T, C, D), the equivalence classes in $T/(C \cup D)$ are also referred to as granules. Specially, equivalence classes in T/C are C-granules and T/D are D-granules. Then, the equivalence class in $T/(C \cup D)$ (or in T/C, T/D) containing t is a granule (or C-granule or D-granule) induced by t, and is denoted by $(C \cup D)(t)$ (or $C(t)$ or $D(t)$).

Definition 5: *Coverset* of granules. Let $cg = C(t)$ be a C-granule induced by transaction t. Its covering set $coverset(cg) = \{t' | t' \in T, (t', t) \in I(C)\}$.

The covering sets of other kinds of granules can be defined similarly. Then there are the following theorem based on these definitions.

Theorem 2. Let granule $g = cg \wedge dg$, where cg is a C-granule and dg is a

D-granule, then we have $coverset(g) = coverset(cg) \cap coverset(dg)$.

Proof: Let g be induced by t such that $g = (C \cup D)$, then we have $cg = C(t)$ and $dg = D(t)$. Assume transaction $t' \in T$. Because:

$$(t', t) \in I(C \cup D) \Leftrightarrow (t', t) \in I(C) \; and \; (t', t) \in I(D).$$

we have

$$t' \in coverset(g) \Leftrightarrow (t', t) \in I(C \cup D) \Leftrightarrow (t', t) \in I(C)$$

and

$$(t', t) \in I(D) \Leftrightarrow t' \in (coverset(cg) \cap coverset(dg)). \; \square$$

For example, using Tables 3.3, 3.4 and 3.5, we have:

$$\begin{aligned} g_1 &= (a_1 = 1 \wedge a_2 = 1 \wedge a_3 = 0 \wedge a_4 = 0 \wedge a_5 = 0 \wedge a_6 = 0 \wedge a_7 = 0) \\ &= ((a_1 = 1 \wedge a_2 = 1 \wedge a_3 = 0 \wedge a_4 = 0 \wedge a_5 = 0) \wedge (a_6 = 0 \wedge a_7 = 0)) \\ &= C(t_1) \wedge D(t_1) \\ &= cg_1 \wedge dg_1. \end{aligned}$$

Therefore, from Theorem 2, we have:

$$\begin{aligned} coverset(g_1) &= coverset(cg_1 \wedge dg_1) \\ &= coverset(cg_1) \cap coverset(dg_1) \\ &= \{t_1, t_5, t_6\} \cap \{t_1\} \\ &= \{t_1\}, \end{aligned}$$

which is the same result as shown in Table 3.3.

Definition 6: Derived decision pattern. Let X be a pattern and $B \subseteq V^T$ be a set of attributes. Then X is a *decision pattern* of B if $\exists g \in T/B$, such that $X = \{a_i \in B | a_i(g) = 1\}$. X is also called the derived decision pattern of granule g.

Theorem 3. Let (T, V^T, C, D) be a decision table and $C \cup D = V^T$. We have

(1) $coverset(C(t)) \supseteq coverset((C \cup D)(t))$, for all $t \in T$.

(2) The derived decision pattern of every granule $g \in T/(C \cup D)$ is a closed pattern.

(3) Max closed patterns are decision patterns of $(C \cup D)$.

Proof: (1) is obvious according to **Definition 5**.

For (2), let X be the derived pattern of g, that is, $X = \{a_i \in C \cup D | a_i(g) = 1\}$. From the definition of granules, there is a transaction $t_0 \in coverset(g)$ such that $X = \{a_i \in C \cup D | a_i(t_0) = 1\}$, that is $t_0 \in coverset(X)$.

Given an item $a \in itemset(coverset(X))$, according to Definition 2, we have $a \in t$ for all $t \in coverset(X)$–that is, $a \in t_0$; hence, $a \in X$ (notice: $X = \{a_i \in$

$C \cup D | a_i(t_0) = 1\} = \{a_i \in V^T | a_i(t_0) = 1\})$.

Therefore, $closure(X) = itemset(coverset(X)) \subseteq X$.

We also have $X \subseteq closure(X)$ from Theorem 1; hence, we have $X = closure(X)$.

For (3), assuming X be a max closed pattern, it is obvious that $X \subseteq t$ for all $t \in coverset(X)$. If there is a transaction $t \in coverset(X)$ such that $X \subset t$, And if Y is a largest t, such that $X \subset t$, then we have $X \subset Y$.

For any $a \in closure(Y)$, we can say $a \in Y$; otherwise, let $Z = Y \cup \{a\}$, and we have:

$X \subset Y \subset Z$;

$\exists t_0 \in coverset(Y) \Rightarrow Z \subseteq \{a_i \in V^T | a_i(t_0) = 1\}$.

Therefore, $X \subseteq \{a_i \in V^T | a_i(t_0) = 1\}$ and then $t_0 \in coverset(X)$. This conflicts with the assumption of the largest Y. Thus, $a \in Y$ and $Y \supseteq closure(Y)$.

According to Theorem 2, we have $closure(Y) \supseteq Y$. Then Y is closed pattern. This is a conflict to the assumption. Therefore, $X = t$ for $t \in coverset(X)$. That is, X is the derived decision pattern of granule$(C \cup D)(t)$.□

For example, from Table 3.3, we have

$$g_1 = (a_1 = 1 \wedge a_2 = 1 \wedge a_3 = 0 \wedge a_4 = 0 \wedge a_5 = 0 \wedge a_6 = 0 \wedge a_7 = 0).$$

In addition, in Table 3.6, there is a pattern of $\{a_1, a_2\}$. This pattern is thus the derived decision pattern of g_1, and is a max closed pattern.

Theorem 4. Let (T, V^T) be and information table and let X be a closed pattern, but not a max closed pattern. There exists a $C \subseteq V^T$ and a *C-granule*

cg such that $C \subset V^T$ and X is derived decision pattern of cg.

Proof: Let $C = \{a|a \in X\} \subseteq V^T$. Because X is a closed pattern, there is a transaction $t \in T$ such that $X \subseteq t$. Let $cg = C(t)$, the induced *C-granule* by t, then $X = \{a_i \in C|a_i(cg) = 1\}$–that is, X is the derived decision pattern of cg, according to Definition 6.

It is also can be observed that there is a closed pattern Y such that $X \subset Y$, since X is not a max closed pattern. It is obvious that $C = \{a|a \in X\} \subset \{a|a \in Y\} \subseteq V^T$.□

For example, for pattern $\{a_6\}$ in Table 3.6, the covering transactions are t_2, t_3, t_4, t_5, t_6. From Table 3.5, we have $C(t_2) = C(t_3) = C(t_4) = dg_2$ and $C(t_5) = C(t_6) = dg_3$, such that pattern $\{a_6\}$ is a derived pattern of dg_2.

Through the formal discussion about the relationships between granules and patterns above, a conclusion can be drawn as follows: a decision rule that corresponds to a granule can also be interpreted as a decision pattern. First, the discussion on Theorem 3 proves that decision rules or decision patterns and max closed patterns mutually correspond to each other. Second, small closed patterns can be interpreted as smaller granules correspondingly, which is indicated in Theorem 4.

These results imply that granules are significant abstractions of knowledge in a database because they can prune noises that are possibly carried by the non-closed patterns. Moreover, they also indicate that decision tables are not adequate to manage granules because decision tables are incomplete for interpreting all closed

patterns; thus, decision tables are incapable of handling the associations between granules that are smaller than the decision rules. These findings lead to the introduction of the multi-tier structure to overcome the limitations of decision tables.

3.6 Multi-tier structure of granules

As illustrated in [49], decision tables can be described as two-tier structures that seek to efficiently discover the relationships between condition granules and decision granules. However, when the number of attributes becomes larger, the two-tier structure is inefficient. Further, the two-tier structure is incapable of describing decision rules with different sizes. In addition, there is no mechanism to filter out meaningless rules when using only the two-tier structure. To resolve these issues, a multi-tier structure was proposed to represent granules in multiple tiers. In addition, this multi-tier structure was further developed to efficiently discover the associations between granules of different sizes in different tiers. The multi-tier solution includes the concept of multi-tier structures, as well as the concept of general rules of decision rules (rules with shorter antecedents) that are used to clarify the meanings of meaningless items in granule mining.

In order to describe the associations between the granules in different sizes, the condition attributes can be further divided into some sub-categories according to the user constrains and the hierarchy structures of the data. Let C_i and C_j be two subsets of condition attributes, where C_i and C_j satisfy the condition

C_i-granule	a_1	a_2	coverset
$cg_{i,1}$	1	1	$\{t_1, t_5, t_6\}$
$cg_{i,2}$	0	0	$\{t_2, t_3, t_4\}$

Table 3.7: C_i-granules

C_j-granule	a_3	a_4	a_5	coverset
$cg_{j,1}$	0	0	0	$\{t_1, t_5, t_6\}$
$cg_{j,2}$	1	1	0	$\{t_2\}$
$cg_{j,3}$	1	1	1	$\{t_1, t_4\}$

Table 3.8: C_j-granules

that $C_i \cap C_j = \emptyset$ and $C_i \cup C_j = C$. For the condition granules in Table 3.4, with the division that makes $C_i = \{a_1, a_2\}$ and $C_j = \{a_3, a_4, a_5\}$, the resulting smaller granules are shown in Tables 3.7 and 3.8. These smaller granules are called C_i-granules and C_j-granules respectively.

For a multi-tier structure, it is formally denoted as the pair of ($\mathbb{H}$, $\mathbb{A}$), where $\mathbb{H}$ is a set of granule tiers and $\mathbb{A}$ is a set of association mappings that represent the associations between the granules in different tiers.

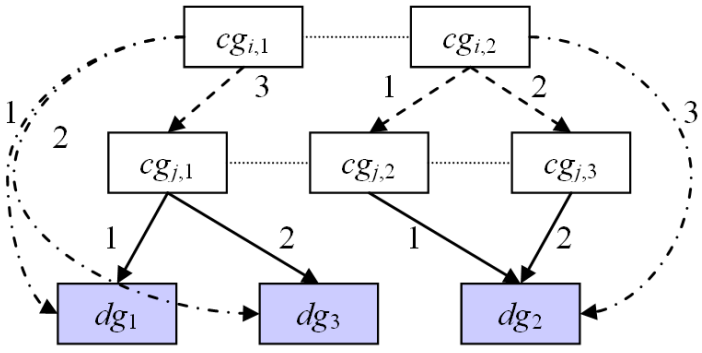


Figure 3.4: An example of a multi-tier structure

Figure 3.4 depicts an example of a three-tier structure for the possible granules in Tables 3.5, 3.7 and 3.8. This three-tier structure is generated by dividing the

C-granules into C_i granules and C_j granules, which are the first two levels in Figure 3.4. Such a three-tier structure is denoted as $\mathbb{H} = \{C_i, C_j, D\}$, where the C_i tier contains the C_i-granules = $\{cg_{i,1}, cg_{i,2}\}$, the C_j tier contains the C_j-granules = $\{cg_{j,1}, cg_{j,2}, cg_{j,3}\}$ and the D tier contains the D-granules = $\{dg_1, dg_2, dg_3\}$.

The three-tier structure shown in Figure 3.4 also contains three types of association mappings that are depicted as the arrows between the granules in the picture. The set of association mappings are denoted as $\mathbb{A} = \{\Gamma_{cd}, \Gamma_{ij}, \Gamma_{id}\}$. The association mapping Γ_{cd}, Γ_{ij}, Γ_{id} represent the linkages between the C-granules and D-granules, C_i-granules and C_j-granules, and C_i-granules and D-granules, respectively. These association mappings are the entities where the decision rules can be generated from.

Given a C-granule cg_k and C_i-granule $cg_{i,x}$, there are three sets of association mappings, including $\Gamma_{CD}(cg_k)$, $\Gamma_{ij}(cg_{i,x})$ and $\Gamma_{id}(cg_{i,x})$, respectively. $\Gamma_{cd}(cg_k)$ consists of all possible associations between granule cg_k and the D-granules, which includes the links starting from cg_k to every related granule in the D tier, and the strength of these links. Similarly, for $\Gamma_{ij}(cg_{i,x})$, it includes all possible associations between $cg_{i,x}$ and C_j-granules that comprise the links from $cg_{i,x}$ to granules in tier C_j. Finally, the mapping $\Gamma_{id}(cg_{i,x})$ includes all possible associations between the granule $cg_{i,x}$ and D-granules.

For example, using Tables 3.4, 3.7, and 3.8, we have:

$$cg_1 = cg_{i,1} \wedge cg_{j,1}$$

$cg_2 = cg_{i,2} \wedge cg_{j,2}$

$cg_3 = cg_{i,2} \wedge cg_{j,3}$

Using Tables 3.3 and 3.5, we also have:

$\Gamma_{cd}(cg_1) = \{(dg_1, 1), (dg_3, 2)\}$

$\Gamma_{cd}(cg_2) = \{(dg_2, 1)\}$

$\Gamma_{cd}(cg_3) = \{(dg_2, 2)\}$

The *link-strength* between condition granule cg_k or $cg_{i,x}$ and decision granule dg_z is defined as follow:

$$\begin{aligned} \textit{link-strength}(cg_k, dg_z) &= |coverset(cg_k \wedge dg_z)| \\ (\text{or } \textit{link-strength}(cg_{i,x}, dg_z) &= |coverset(cg_{i,x} \wedge dg_z)|) \end{aligned} \tag{3.4}$$

The link-strength equals to the number of transactions that satisfy the granule "$cg_k \wedge dg_z$" (or "$cg_{i,x} \wedge dg_z$").

Then the expression of '$cg_k \rightarrow dg_z$' is called a 'decision rule' where cg_k is the antecedent part and dg_z is the consequent part of the rule. The calculation of the support of this decision rule is as the follow:

$$sup = \frac{|coverset(cg_k \wedge dg_z)|}{|T|} = \frac{1}{N} \textit{link-strength}(cg_k, dg_z) \tag{3.5}$$

where $N = |T|$, the total number of transactions.

And the confidence of the decision rule is as follow:

$$conf = \frac{|coverset(cg_k \wedge dg_z)|}{|coverset(cg_k)|} = \frac{1}{|coverset(cg_k)|} \textit{link-strength}(cg_k, dg_z). \tag{3.6}$$

	Link between granules	link-strength
C-granules and D-granules	$cg_{i,1} \wedge cg_{j,1} \rightarrow dg_1$	1
	$cg_{i,1} \wedge cg_{j,1} \rightarrow dg_3$	2
	$cg_{i,2} \wedge cg_{j,2} \rightarrow dg_2$	1
	$cg_{i,2} \wedge cg_{j,3} \rightarrow dg_2$	2
C_i-granules and C_j-granules	$cg_{i,1} \rightarrow cg_{j,1}$	3
	$cg_{i,2} \rightarrow cg_{j,2}$	1
	$cg_{i,2} \rightarrow cg_{j,3}$	2
C_i-granules and D-granules	$cg_{i,1} \rightarrow dg_1$	1
	$cg_{i,1} \rightarrow dg_3$	2
	$cg_{i,2} \rightarrow dg_2$	3

Table 3.9: Associations between granules

Table 3.9 lists the possible links and their link-strengths shown in Figure 3.4, where cg_k can also be represented in the form '$cg_{i,x} \wedge cg_{i,y}$' in the three-tier structure shown in Figure 3.4.

The introduction of the multi-tier structure has shown that there are rules with different premises contained in the multi-tier structure. This leads to the possibility of discussing general rules with shorter premises. Moreover, with the concept of general rules, it is possible to define the term 'meaningless' in the multi-tier structure for a decision rule on selected tiers. The following discussion presents the formal definitions of general rules and meaningless rules.

Definition 7. Let cg_k be a C-granule and $cg_k = cg_{i,x} \wedge cg_{k,y}$. Then rule '$cg_{i,x} \rightarrow dg_z$' is a general rule of rule '$cg_k \wedge dg_z$'.

Definition 8. Let cg_k be a *C-granule* and $cg_k = cg_{i,x} \wedge cg_{k,y}$. Then rule '$cg_k \rightarrow dg_z$' is a meaningless rule if its confidence is less than or equal to the confidence of its general rules.

The rationale of the definition of meaningless rules is analogous to the defi-

nition of interesting rules, where $X \rightarrow Y$ is an interesting rule if the conditional probability $P(Y|X)$ is greater than $P(Y)$. If adding a piece of extra evidence to a premise obtains a weak conclusion, the piece of evidence can be defined as meaningless.

3.7 Association mappings for efficient rule generation

The last section has presented a three-tier structure ($\mathbb{H}$, $\mathbb{A}$), where $\mathbb{H} = \{C_i, D_j, D\}$, $C_i \cup C_j = C$ and $C_i \cap C_j = \emptyset$, and $\mathbb{A} = \{\Gamma_{CD}, \Gamma_{ij}, \Gamma_{id}\}$. Association mappings are the entities used to describe the association relationships between granules in different tiers. They can be used to enumerate all the association rules between the associated granules. In usual cases, there are many possible pairs of (C_i, C_j), such that $C_i \cup C_j = C$ and $C_i \cap C_j = \emptyset$, and C_i or C_j or both can be divided into smaller granule sets. Therefore, it is necessary to use the derived association mappings (such as Γ_{id}) for efficient rule generation in the multi-tier structures.

The very important regulation for obtaining association mappings is the completeness. That is, the association rules that are discovered from the association mappings should be the same as the rules obtained from the original database under the corresponding user constraints. To achieve this purpose, the first step is to formalise a basic association mapping in the decision tables. The next step

is to develop methods to derive other association mappings between the granules in different tiers based on the basic association mappings.

3.7.1 The Basic association mapping

The basic association mappings are the associations that link the condition granules to the corresponding decision granules. As defined in the previous sections, let $U = T/V^T$, $U_C = T/C$ and $U_D = T/D$ be two finite and non-empty sets. Then the elements of U_C and U_D are the condition granules and decision granules, respectively.

The relationships between condition granules (or decision granules) and transactions can also be described in formal concept analysis (also see [117] for this definition) by using a relation R_C (R_D) between U_C (U_D) and T. For a pair of transaction, t, and granule, g, such that $t \in T$ and $g \in U_C$, if $(t,\ g) \in R_C$–also written as tR_Cg–then we can say that g is induced by t, or t has the property g.

For example, using the examples in Table 3.3 and 3.4, we have:

$$U_C = \{cg_1, cg_2, cg_3\},$$

and

$$R_C = \{(t_1, cg_1), (t_5, cg_1), (t_6, cg_1), (t_2, cg_2), (t_3, cg_3), (t_4, cg_3)\}.$$

Given a granule $g \in U_C$, its covering set $coverset(g) = \{t \in T | tR_Cg\}$.

Let $g_1 \in U_C$ and $g_2 \in U_D$, based on the Eq. 3.4 and Theorem 2, then we have:

$$\begin{aligned} \textit{link-strength}(g_1, g_2) &= |coverset(g_1 \wedge g_2)| \\ &= |coverset(g_1) \cap coverset(g_2)| \\ &= |\{t \in T | tR_C g_1 \text{ and } tR_D g_2\}|. \end{aligned}$$

The association between C-granules and D-granules can be described as a basic association mapping, Γ_{cd}, such that $\Gamma_{cd}(g)$ is a set of D-granule link-strength pairs for all $g \in U_C$. Formally, Γ_{cd} is defined as follows:

$$\Gamma_{cd} :: U_C \rightarrow 2^{U_D \times I} \tag{3.7}$$

which satisfies:

$$\Gamma_{cd}(g) = \{(dg, \textit{link-strength}(g, dg)) | dg \in U_D, \{t \in T | tR_C g \textit{ and } tR_D dg\} \neq \emptyset\} \tag{3.8}$$

for all granules $g \in U_C$, where I is the set of all integers.

For example, using the granules in Figure 3.4 and Table 3.9, we have:

$$\Gamma_{cd}(cg_1) = \Gamma_{cd}(cg_{i,1} \wedge cg_{j,1}) = \{(dg_1, 1), (dg_2, 2)\}.$$

From the above equation, there are two decision rules can be derived as follows:

$$cg_{i,1} \wedge cg_{j,1} \rightarrow dg_1 \ (\textit{link-strength} = 1)$$

and

$$cg_{i,1} \wedge cg_{j,1} \rightarrow dg_2 \ (\textit{link-strength} = 2).$$

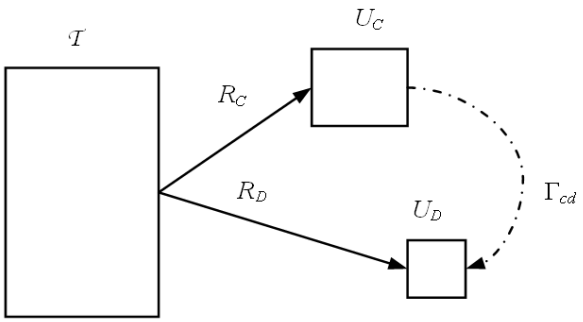


Figure 3.5: Relations for the basic association mapping

Figure 3.5 illustrates the relations between granules and transactions and relations between C-granules and D-granules, where relations R_C and R_D are used to compress transactions, T, into condition granules, U_C, and decision granules, U_D, respectively, and the basic association mapping, Γ_{cd}, is used to indicate the association relationship between condition granules and decision granules.

Based on the basic association mappings, it is obvious that the supports and confidence of the decision rules can be easily calculated. Let $g_1 \in U_C$, $g_2 \in U_D$, and '$g_1 \rightarrow g_2$' be a decision rule, its support and confidence can be derived from the following equations:

$$\textit{sup}(g_1 \rightarrow g_2) = \frac{1}{N}\textit{link-strength}(g_1, g_2) = \frac{1}{N} \sum_{(g_2, ls) \in \Gamma_{cd}(g_1)} ls \tag{3.9}$$

$$\textit{conf}(g_1 \rightarrow g_2) = \frac{\textit{link-strength}(g_1, g_2)}{|\textit{coverset}(g_1)|} = \frac{\sum_{(g_2, ls) \in \Gamma_{cd}(g_1)} ls}{\sum_{(g, ls) \in \Gamma_{cd}(g_1)} ls} \tag{3.10}$$

3.7.2 Derived association mappings

The very interesting property of multi-tier structure is that not only the basic association mappings cab be generated from it, but also many other association mappings can be derived without using the original transactions. This property is significant for time complexities of rule generations.

There are different methods to derived different kinds of association mappings. There are methods for deriving the association mappings between the divided condition granules and methods for association mappings between the condition granules in upper tier and the decision granules. To simplify the process of deriving, the method for deriving association mapping, Γ_{ij}, based on basic association mappings, Γ_{cd}, will be introduced first. The association mapping Γ_{ij} is a set of C_j-granules integer pairs which satisfies:

$$\Gamma_{ij} :: U_i \rightarrow 2^{U_j} \times I \tag{3.11}$$

such that

$$\Gamma_{ij}(g_i) = \{(g_j, \textit{link-strength}(g_i, g_j))|g_j \in U_j, \{t \in T|tR_ig_i \textit{ and } tR_jg_j\} \neq \emptyset\}$$

for all granules $g_i \in U_i$, where $C_i \cup C_j = C$, $C_i \cap C_j = \emptyset$, $U_i = T/C_i$ (set of C_i-granules), $U_j = T/C_j$ (set of C_j-granules), and R_i and R_j are relations between U_i and T, and U_j and T, respectively.

The second deriving method is for the association mapping, Γ_{id}, between C_i-

granules and D-granules based on the association mappings, Γ_{ij} and Γ_{cd}, which satisfies:

$$\Gamma_{id} :: U_i \rightarrow 2^{U_D} \times I \tag{3.12}$$

such that

$$\Gamma_{ij}(g_i) = \{(dg, \textit{link-strength}(g_i, dg))|dg \in U_D, \{t \in T|tR_ig_i \textit{ and } tR_Ddg\} \neq \emptyset\}$$

for all granules $g_i \in U_i$.

For example, from Tables 3.7 and 3.8, we have:

$$U_i = \{cg_{i,1}, cg_{i,2}\} \;\; \textit{and} \;\; U_j = \{cg_{j,1}, cg_{j,2}, cg_{j,3}\}.$$

We also have:

$$R_i = \{(t_1, cg_{i,1}), (t_5, cg_{i,1}), (t_6, cg_{i,1}), (t_2, cg_{i,2}), (t_3, cg_{i,2}), (t_4, cg_{i,2})\}$$

and

$$R_j = \{(t_1, cg_{j,1}), (t_5, cg_{j,1}), (t_6, cg_{j,1}), (t_2, cg_{j,2}), (t_3, cg_{j,3}), (t_4, cg_{j,3})\}.$$

Figure 3.6 illustrates the relations between these association mappings. As shown in this figure, the set of condition attributes are split into two sets–C_i

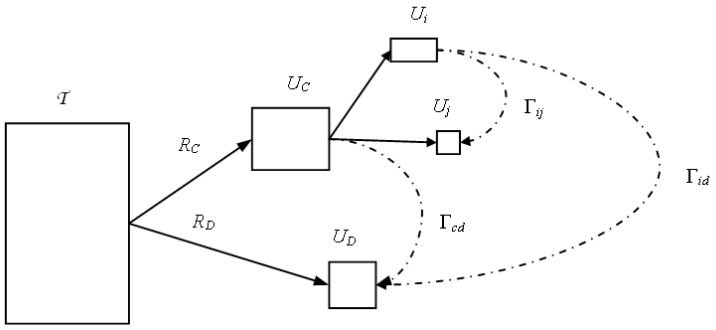


Figure 3.6: Relations for derived association mappings

and C_j–and the C-granules(U_C) are also correspondingly compressed into C_i-granules(U_i) and C_j-granules(U_j). As defined previously, Γ_{cd} is used to describe the association relationship between U_C and U_D. Association mapping, Γ_{ij}, is used to describe the association relationship between U_i and U_j, and association mapping, Γ_{id}, is used to describe the association relationship between U_i and U_D.

3.8 Summary

This chapter first presented an overview of the system to implement the proposed approach. The system retrieves the data from the data warehouse fact table, and transforms the data into an information table. Following this, it generates the granules and constructs the multi-tier structure according to the user's constraints. It also generates the association and presents the results to the user, according to the user's requests.

This chapter also discussed the patterns, association rules and granules, using the same example transactions. It first described how the frequent patterns

and closed patterns were obtained from the table, and the association rules generated from these patterns. It then described the condition granules, decision granules and decision rules generated from the same data. This discussion illustrated the difference between granules and patterns. The first difference is that granule mining is a one-phase process, while association rule mining has two stages. Moreover, granules have more semantic meaning than do patterns because granules are predicates. The third difference is that granules can directly describe multiple values, and the final difference is that granules can be divided into smaller granules to reduce the number of granules. However, the decision table is inefficient for a large number of attributes and cannot handle granules of different sizes. These factors motivated the development of our approach.

Section 3.4 presented the interpretation of granules in terms of patterns in order to interpret decision rules in terms of association rules. A set of definitions was presented to illustrate the relationships between patterns and granules, including definitions for the coverset of itemset, itemset of a set of transactions, closure, coverset of granules and some theorems deduced from these definitions. From the discussion of these relationships, we concluded that a decision rule corresponding to a granule can be interpreted as a decision pattern from two aspects. First, decision patterns and max closed patterns mutually correspond to each other. Second, small closed patterns can be interpreted as small granules. This conclusion implies that granules can prune noises in non-closed patterns and can completely interpret all closed patterns, which decision tables are unable to

achieve. This led to the development of the multi-tier structures.

Section 3.5 presented an extended description of multi-tier structure construction and meaningless rules, which was proposed in [48]. The condition attributes of a two-tier structure can be divided into smaller groups to describe the associations between granules of different sizes. A multi-tier structure consists of a set of granule tiers and a set of association mappings, and is denoted as ($\mathbb{H}$, $\mathbb{A}$). A three-tier structure can be constructed by dividing the C-granule into two smaller subsets–C_i and C_j granules–such that the three-tier structure consists of $\mathbb{H} = \{C_i, C_j, D\}$ and $\mathbb{A} = \{\Gamma_{cd}, \Gamma_{ij}, \Gamma_{id}\}$. For every granule in a tier, the mappings include all associations of the granules in the corresponding tier of the mapping. Following this, the support and confidence of the decision rule can be calculated through the link-strength of the mappings. Moreover, with association mappings, it is possible to have general rules with shorter premises, and–with the general rule–the 'meaningless rule' (of having less or the same confidence of the general rule) can be defined.

This chapter also presented a discussion of the basic and derived association mappings of the three-tier structure. To achieve completeness, the association rules that are discovered from the association mappings should be the same as the rules obtained from the original database, under the corresponding user constraints. Therefore, we first formalised the basic association mappings in the decision table. Following this, we used the basic association mappings to derive other association mappings between the granules in different tiers. The transactions

were compressed into condition and decision granules through the corresponding relations, and the basic mappings indicated the association relations between the granules of these two tiers. Then, for a condition granule, its basic association mapping is a set of D-granule and link-strength pairs. There are two types of derived association mappings. The first is the association mappings between the smaller condition granules, while the second is the association mappings between the smaller condition granules and decision granules. To generate the derived mappings, the condition granule, C-granule, was first compressed into smaller condition granules, such as C_i and C_j. Following this, we used the relations between association mappings to produce methods for calculating the derived association mappings. For the association mappings between condition granules, we used the concept of the generalised granule to generate the derived mappings. The other type of derived mappings was generated by conducting composition of the basic association mappings and the first type of derived association mappings.

Chapter 4

Support estimation

Support estimation was originally proposed to provide a method to restore support for the patterns that are summarised into a limited number of profiles or compressed representation. Usually a pattern's support cannot be obtained directly from the profiles or compressed representatives. Thus, the support of a given pattern can only be estimated through the corresponding restoration calculation by using the information stored in the profiles or compressed representatives. This chapter first introduces profile-based pattern summarisation and support estimation by profile in more detail. It then presents the estimated support calculations for the granule mining approach.

The reasons that we use the term 'estimation' in this study are as follows. First, the estimation is proposed for pattern summarization and we used this

idea in our research. Second, we have proposed a precise method and an approximate method for support estimation as well. Such that, we keep using the term estimation.

4.1 Support estimation for summarization

As described in the literature review, profile based pattern summerization is propose in [112]. It summarizes the closed frequent patterns into a certain number of pattern profiles to reduce the overwhelming number of discovered patterns. Then, to obtain the support for a given pattern, it approximately calculated the estimated support using these profiles.

In a transaction dataset D, let α_1, α_2, ... , α_l be a set of patterns and D' = $\bigcup_{1 \leqslant i \leqslant l} D_{\alpha_i}$, where D_{α_i} is the coverset of α_i. Then the profile, M, is a triple $\langle P, \phi, \rho \rangle$, where P is the probability distribution vector of the items in this profile. The item probability is the percentage of transactions in D' containing the item. ϕ is called master pattern, which is the union of α_1, α_2, ... , α_l, and ρ is the support of the profile, which equals to $\frac{|D'|}{|D|}$.

With the profiles, the estimated support can be calculated for the given patterns. The calculation equation is defined as below:

$$\hat{s}(\alpha_k) = max_{\mathrm{m}}(s(\mathrm{M}) \times \prod_{\mathrm{o_i} \in \alpha_{\mathrm{k}}} \mathrm{p_M}(\mathrm{x_i} = 1)) \tag{4.1}$$

For a given pattern α_k, its support is decided by the product of its items'(σ_i)

probabilities ($p(x_i{=}1)$) in the profile, and the support of the profile's master pattern ρ ($s(\mathrm{M})$). Since it is possible that the pattern to be the subset of several master patterns. Therefore, for every profile that contains the pattern, there is an estimated support of the pattern can be calculated from this profile. Among the candidates of estimated support, the maximal one is selected as the estimated support for the given pattern.

The accuracy of the estimated support can be measured by the average relative error between the estimated support and original support. This measure is also called the 'restoration error rate'. For profile-based pattern summarisation, the value of the restoration error rate depends on the number of profiles. Specifically, let the number of profiles be K; then, the larger K is, the smaller the restoration error rate becomes. Thus, K is set by an optimal method–that is, if the restoration error rate drops significantly when the profile numbers increase from K-1 to K, then K is the optimal number of profiles.

4.2 Support estimation for granules

In terms of the multi-tier structure of granules, the estimated support is calculated through the granules and association mappings. The estimated support can be calculated by the granule support if the given pattern is derived by the granules in only one tier. Otherwise, the estimated support can be calculated through the link-strength of the association mappings between the granules containing the pattern. In some circumstances, the estimated support calculated through the

multi-tier structure can achieve a zero restoration error rate.

There are several different calculation methods to compute the estimated support from the multi-tier structure of granules according to the definition of the current multi-tier structure and the tiers of granule containing the given pattern.

The first case is to estimate the support for a pattern with a decision table. Let G be the decision table of a information table(T, V^T), then the estimated support is calculated solely through sum of support of the granules(decision rules) containing the pattern. The equation to calculate the estimated support for a given pattern, α, is as follow:

$$\hat{s}_1(\alpha, G) = \frac{\sum\limits_{g \in G, \alpha \subseteq g} \sup(g)}{\sum\limits_{g_i \in G} \sup(g_i)} = \frac{\sum\limits_{g \in G, \alpha \subseteq g} \sup(g)}{|T|}$$

The second case is calculating the estimated support with a two-tier structure. For a two-tier structure, let CG be the set of C-granules and DG be the set of D-granules. A pattern, α, can be divided into two patterns, α_1 and α_2, such that $\alpha_1 = \alpha \cap C$ and $\alpha_2 = \alpha \cap D$, respectively. Then, the estimation support is calculated as the summary of granules support if pattern α only derives from the granules in one tier. Additionally, it can be calculated through the link-strength of the association mappings between the granules that contain α_1 and α_2. The equations for the estimated support calculation is as follow:

$$\hat{s}_2(\alpha, CG, DG) = \begin{cases} \frac{1}{|T|} \sum\limits_{g \in CG, \alpha_1 \subseteq g} \sup(g) = \hat{s}_1(\alpha_1, CG) & \text{if } \alpha_2 = \emptyset \\ \frac{1}{|T|} \sum\limits_{g \in DG, \alpha_2 \subseteq g} \sup(g) = \hat{s}_1(\alpha_2, DG) & \text{if } \alpha_1 = \emptyset \\ \frac{1}{|T|} \sum\limits_{\alpha_1 \subseteq g_1 \in CG, \alpha_2 \subseteq g_2 \in DG} \mathit{lstrengh}(g_1 \rightarrow g_2) & \text{otherwise} \end{cases} \tag{4.2}$$

For other cases with multi-tier structures that have three or more tiers, the estimated support can be calculated using different methods, depending on the number of tiers of granules from which the given pattern has been derived. There are three categories of calculation method for estimated support in the multi-tier structure. The first case is that the given pattern only derives from the granules in one tier, and the support can be calculated using the supports of the granules in the corresponding tier. The second case is for patterns that derive from the granules of two tiers. The calculation for such cases can use the link-strength of the association mappings between the two granules to obtain the support. This calculation can be performed directly using the current multi-tier structure. The third method is for patterns that derive from granules in three or more tiers. To acquire the estimated support for such patterns, the calculation can either use the mapping information from the two-tier structure to compute the support through Eq.(4.2), or can perform an approximate calculation to estimate the support.

To demonstrate the estimated support calculation from the multi-tier structure, we use a three-tier structure as an example to illustrate the calculation

details. Let a three-tier structure be $\mathbb{H} = \{C_i, C_j, D\}$, containing three sets of granule that are C_iG, C_jG and DG. Then, a pattern α can be divided into three patterns: $\alpha_1 = \alpha \cap C_iG$, $\alpha_2 = \alpha \cap C_jG$ and $\alpha_3 = \alpha \cap DG$. If only one of α_1, α_2 or α_3 is non-empty, then the support is calculated through the sum of support of only one set of granules, as follow:

$$\hat{s}_3(\alpha, C_iG, C_jG, DG) = \begin{cases} \frac{1}{|T|} \sum\limits_{g \in C_iG, \alpha_1 \subseteq g} \sup(g) = \hat{s}_1(\alpha_1, C_iG) \text{ if } \alpha_2,\ \alpha_3 = \emptyset \\ \frac{1}{|T|} \sum\limits_{g \in C_jG, \alpha_2 \subseteq g} \sup(g) = \hat{s}_1(\alpha_2, C_jG) \text{if } \alpha_1,\ \alpha_3 = \emptyset \\ \frac{1}{|T|} \sum\limits_{g \in DG, \alpha_3 \subseteq g} \sup(g) = \hat{s}_1(\alpha_3, DG) \text{ if } \alpha_1,\ \alpha_2 = \emptyset \end{cases} \quad (4.3)$$

For the second case, if one of α_1, α_2 and α_3 is empty, then the support is calculated using the link-strength of the association mappings of $\Gamma_{i,j}$, $\Gamma_{i,d}$ or $\Gamma_{j,d}$, as follow:

$$\hat{s}_3(\alpha, C_iG, C_jG, DG) = \begin{cases} \frac{1}{|T|} \sum\limits_{\alpha_1 \subseteq g_1 \in C_iG, \alpha_2 \subseteq g_2 \in C_jG} \mathit{lstrengh}(g_1 \rightarrow g_2) \text{if } \alpha_3 = \emptyset \\ \frac{1}{|T|} \sum\limits_{\alpha_1 \subseteq g_1 \in C_iG, \alpha_3 \subseteq g_3 \in DG} \mathit{lstrengh}(g_1 \rightarrow g_3) \text{if } \alpha_2 = \emptyset \\ \frac{1}{|T|} \sum\limits_{\alpha_2 \subseteq g_2 \in C_jG, \alpha_3 \subseteq g_3 \in DG} \mathit{lstrengh}(g_2 \rightarrow g_3) \text{if } \alpha_1 = \emptyset \end{cases} \quad (4.4)$$

Finally, if none of α_1, α_2 or α_3 is empty, then it is a case of the third category. In this case, the support is calculated through the mappings of $\Gamma_{c,d}$. In order

to use these mappings, the division of α needs to be modified. That is, let $\alpha_1 \cup \alpha_2 = \alpha \cap CG$, where $CG = C_i \cup C_j$, such that the support can be obtained by using a modified version of Eq.(4.2). The equation used for this calculation is as follow:

$$\begin{aligned} \hat{s}_3(\alpha, C_iG, C_jG, DG) = \tfrac{1}{|T|} \sum_{Cond} lstrengh((g_1 \wedge g_2) \rightarrow g_3) \\ Cond : \alpha_1 \subset g_1 \in C_iG, \alpha_2 \subset g_2 \in C_jG, \\ C_iG \cup C_jG = CG, \alpha_3 \subseteq g_3 \in DG \end{aligned} \tag{4.5}$$

Otherwise, $\hat{s}_3$ can be calculated using the following calculation:

$$\frac{1}{|T|} \sum \min\{lstrength(g_1 \rightarrow g_2),\ lstrength(g_1 \rightarrow g_3),\ lstrength(g_2 \rightarrow g_3)\} \tag{4.6}$$

It should be noted that this calculation is an approximate equation that carries some losses. The formula uses the sum of the minimum link-strength of the three association mappings between the granules of two tiers to approximate the support for the given patterns derived from the granules from all tiers.

Regarding the quality of the estimation, when using the two-tier structure to calculate the estimated support, it can achieve a zero restoration error rate because the two-tier structure is a lossless compression. Further, for multi-tier structures with more than two tiers, it can also achieve a zero error rate when

using only the mappings of granules from two tiers or the calculation is performed via the basic two-tier structure.

Theorem 8. For a given pattern, α, and a multi-tier structure, $\mathbb{H} = \{C, D\}$, the estimated support calculated through $\mathbb{H}$ equals to the original support of α. That is, $\hat{s}(\alpha, \mathbb{H}) = Sup_\alpha$.

Proof. For a pattern, α, let $\alpha = \alpha_1 \cup \alpha_2$, such that $\alpha_1 \cap \alpha_2 = \emptyset$. Then, the support of α, $Sup_\alpha = |coverset(\alpha_1) \cap coverset(\alpha_2)|$.

Assume $\alpha_1 \subset g_1$ and $\alpha_2 \subset g_2$, such that $g_1 \in CG$ and $g_2 \in DG$. This leads to:

$$coverset(\alpha_1) = \bigcup_{g_{1,i} \in CG} coverset(g_{1,i})$$

and

$$coverset(\alpha_1) = \bigcup_{g_{2,i} \in CG} coverset(g_{2,i}).$$

Meanwhile, the link-strength of the association mapping from g_1 to g_2 is:

$$\begin{aligned} lstrength(g_1 \rightarrow g_2) &= |coverset(g_1 \wedge g_2)| \\ &= |coverset(g_1) \cap coverset(g_2)|. \end{aligned}$$

Moreover, we have:

$$\begin{aligned} \sum_{g_1 \in CG, g_2 \in DG} lstrength(g_1 \rightarrow g_2) &= |\bigcup_{g_1 \in CG,\, g_2 DG} (coverset(g_1) \cap (coverset(g_2))| \\ &= |\bigcup_{g_{1,i} \in CG} coverset(g_{1,i}) \cap \bigcup_{g_{2,i} \in CG} coverset(g_{2,i})|. \end{aligned}$$

Therefore, we have $Sup_\alpha = \hat{s}(\alpha, \mathbb{H})$. $\square$

4.3 Summary

This chapter first introduced the support estimation by the profile-based summarisation [112], then presented this study's methods to calculate the estimated support using granules. For the profiles, support is calculated by the support of the master pattern and probability distribution vector of the items in the profile. For patterns contained by more than one profile, the estimated support is the maximum support of the supports, calculated by all profiles containing the pattern. To examine the accuracy of the estimated support, the measure of restoration error rate is used to reflect the average relative error between the estimated support and original support.

For the granule approach, the estimated support is calculated according to how the pattern is deduced by the granules in the multi-tier structure. For patterns only derived from granules in one tier, the estimated support is calculated using the support the granules. For the pattern that is derived from the granules of two tiers, the estimated support is calculated using the link-strength between the granules containing the sub-pattern of the given pattern. If there are three or more tiers in the multi-tier structure, if the given pattern is derived from granules of one or two tiers, the estimated support can be calculated using the above methods. Otherwise, if the pattern is contained by granules in three tiers, the estimated support can either be calculated through a two-tier structure or by using the minimum link-strength of the three mappings that contain the pattern. It should be noted that this calculation is only approximate, while the calculation

using the two-tier structure is precise and can achieve a zero restoration error rate.

Chapter 5

Algorithm

The previous chapter formally defined multi-tier structures. One of the advantages of using multi-tier structures is that they can be used to calculate derived association mappings efficiently based on an initial basic association mapping. This chapter first discusses the algorithm for the construction of basic association mapping. It also presents efficient algorithms for derived association mappings based on the results of Theorems 6 and 7 in the previous chapter.

Moreover, Chapter 4 presented several methods to estimate support in the multi-tier structure. These methods included using different calculations to handle different derivation cases of the given pattern, as well as using precise and approximate calculations based on structures with three or more tiers. Several algorithms have been developed for these calculations. They are illustrated in

this chapter.

5.1 Construction and analysis of the basic mapping

Algorithm 1 describes the main procedure of the construction of the basic association mapping. It goes through the set of transactions and constructs the set of conditional granules (U_C) and the basic association mapping (Γ_{cd}). In step 1, U_C is assigned as an empty set. In step 2, the algorithm goes through the set of transactions. For each transaction, $t \in T$, it can be split into two granules: the condition granule, g_1, and decision granule, g_2 (step 4). If a condition granule (g_1) is not in the current U_C, it is added into U_C and the algorithm sets the condition granule an initial value, $(g_1, 1)$, as its mapping result (step 5 to step 8); otherwise, it updates the existing mapping, $\Gamma_{cd}(g_1)$, for the condition granule (step 9 to step 20). If there is a match for the decision granule g_2, the algorithm adds one to the corresponding association strengths (step 12 to step 16); otherwise, it inserts $(g_2, 1)$ into $\Gamma_{cd}(g_1)$ (step 17 to step 19).

Figure 5.1 shows an example of the basic association mapping, with the input information table is Table 3.2, where $C = \{a_1, a_2, a_3, a_4, a_5\}$ and $D = \{a_6, a_7\}$, and the definitions of cg_i $(i = 1, 2, 3)$ and cg_j $(j = 1, 2, 3)$ can be found in Tables 3.7 and 3.8.

Figure 5.2 illustrates the construction process of these mappings using the

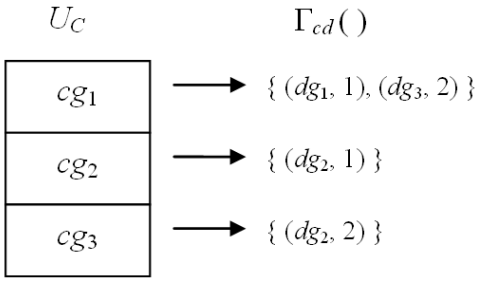


Figure 5.1: An example of the basic association mapping

Algorithm 1, the details are described as follows. For the first transaction, t_1, in Table 3.2, according to the definitions of C-granule, cg, and D-granule, dg, $t_1 = cg_1 \wedge dg_1$. From this definition, we can get the mapping of $\Gamma_{cd}(cg_1) = \{(dg_1,1)\}$. This is the beginning of the process; thus, the collection of granules, U_C, and mappings, Γ_{cd}, is empty. Then, the granule, cg_1, and mapping, Γ_{cg_1}, are inserted to the corresponding collection, respectively. After t_1 is processed, the granules and mappings we have are $U_C = \{cg_1\}$ and $\Gamma_{cd} = \{\Gamma_{cd}(cg_1)$, where $\Gamma_{cd}(cg_1) = \{(dg_1, 1)\}$. Similarly, for transactions t_2 and t_3, granule cg_2 and cg_3 and mapping $\Gamma_{cd}(cg_2) = \{(dg_2, 1)\}$, $\Gamma_{cd}(cg_3) = \{(dg_3, 1)\}$ are generated and inserted into the structure because these granules and mappings do not exist in the structure at this stage. It should be noted that the D-granule generated from t_3 is dg_2, which is the same as the D-granule generated from t_2. However, the mappings generated from these two transactions are different. Thus, the granules and mappings generated from t_2 and t_3 are inserted into the structure as new elements. The structure after processing t_2 and t_3 contains the C-granules, $U_C = \{cg_1, cg_2, cg_3\}$, and the mapping, $\Gamma_{cd} = \{\Gamma_{cd}(cg_1), \Gamma_{cd}(cg_2), \Gamma_{cd}(cg_3)\}$.

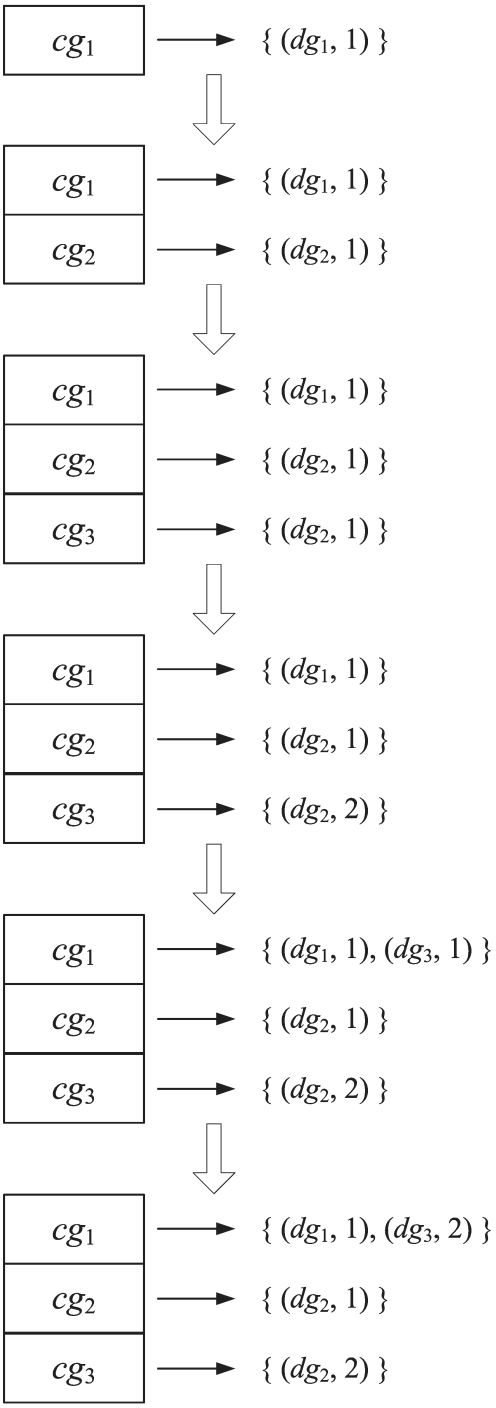


Figure 5.2: Example process of basic mapping generation

When the process proceed to the transaction t_4, the algorithm generates the same granules as transaction t_3, which is granule cg_3. This implies that the

structure contains the C-granule generated from t_4. Moreover, the mapping generated from t_4 is also the same as mapping generated from t_3–that is, the current structure includes this mapping already. Therefore, the algorithm discovers the existing granule and the corresponding mapping to update the current structure accordingly. This operation results in an updated structure that the mapping $\Gamma_{cd}(cg_3)$ is updated to $\Gamma_{cd}(cg_3) = \{(dg_2, 2)\}$.

The process of transaction t_5 is similar to t_4. The C-granule generated is cg_1, which is same as transaction t_1. Hence, the process enters the steps to search the mappings of $\Gamma_{cd}(cg_1)$. The difference between the process of t_5 and t_4 is in this part of the algorithm. The mapping obtained from t_5 is $\Gamma_{cd}(cg_1) = \{(dg_3, 1)\}$, which cannot be found in the structure at this stage. Therefore, the mapping is added to the structure by inserting it into the collection of mapping $\Gamma_{cd}(cg_1)$.

Finally, for transaction t_6, the same granule and mapping as that of t_5 are generated and the structure is updated accordingly. The process then finishes and produces a two tier structure, as shown in Figure 5.1.

The performance analyse of Algorithm 1 is as follows. Assume that the basic operation in Algorithm 1 is the comparison between granules. Let $N = |T|$, $n = |(T/V^T)|$, $n(C) = |U_C|$ and $n(D) = |U_D|$. The time complexity of Algorithm 1 is determined by the 'for' loop in step 2, where checking $(g_1 \in U_C)$ takes $O(n(C))$ for every transaction, and checking $(g = g_2)$ takes $O(n(D))$ if $(g_1 \in U_C)$. Therefore, the time complexity of the algorithm is:

input : (T, V^T), an information table, and sets C and D which satisfy $C \cap D = \emptyset$ and $C \cup D = V^T$.
output: U_C and Γ_{cd}, the set of *C-granules* and the basic association mapping.

```
1  let U_C = ∅ ;
2  foreach t ∈ T do
3      let g_1 ∧ g_2 = t ;
4      /* g_1 is the condition granule and g_2 is the decision granule */
5      if not(g_1 ∈ U_C) then
6          let Γ_cd(g_1) = {(g_2, 1)};
7          let U_C = U_C ∪ {g_1};
8      end
9      else
10         /* updating the existing mapping */
11         let match = false ;
12         foreach (g, s) ∈ Γ_cd(g_1) do
13             if g = g_2 then
14                 s = s + 1, match = true, return;
15             end
16         end
17         if not(match) then
18             let Γ_cd(g_1) = Γ_cd(g_1) ∪ {(g_2, 1)};
19         end
20     end
21 end
```

Algorithm 1: Basic Association Mapping

$$O((n(C) + n(D)) \cdot N) \leq O(N^2) \tag{5.1}$$

because $n(C) \leq N$ and $n(D) \leq N$. Indeed, $n(C) + n(D) < n << N$ in many cases.

input : U_C and Γ_{cd}, the basic association rules.
output: All decision rules.

```
1  let N = 0 ;
2  foreach g ∈ U_C do
3  |   foreach (dg, s) ∈ Γ_cd(g) do
4  |   |   let N = N + s ;
5  |   end
6  end
7  foreach g ∈ U_C do
8  |   let temp = 0;
9  |   foreach (dg, s) ∈ Γ_cd(g) do
10 |   |   let temp = temp + s;
11 |   end
12 |   foreach (dg, s) ∈ Γ_cd(g) do
13 |   |   let sup = s ÷ N, conf = s ÷ temp;
14 |   |   if (sup ≥ min_sup) and (conf ≥ min_conf) then
15 |   |   |   print "g → dg";
16 |   |   end
17 |   end
18 end
```

Algorithm 2: Retrieval of Decision rules

Algorithm 2 describes the process of generating decision rules. Two 'for' loops (in step 2 and step 7) both traverse pairs in $\Gamma_{cd}(g)$ ($g \in U_C$), and the number of pairs is just n; thus, the time complexity of this algorithm is $O(n)$.

Pawlak's method only used decision tables (see [49]). Comparing Algorithms 1 and 2 to Pawlak's method indicates that these algorithms are better than Pawlak's method in terms of time complexities because $n(C) \leq n$ for any $\emptyset \neq C \subseteq V^T$.

5.2 Construction and analysis of the derived mappings

After the construction of the basic association mapping and the set of condition granules, users can further split the condition attributes into any two categories– C_i and C_j–according to what users want, where $C_i \cap C_j = \emptyset$, and $C_i \cup C_j = C$.

To reduce the time complexities for calculating the association mapping between a small set of condition attributes and the condition attributes, we discuss two kinds of derived association mappings between C_i-granules and C_j-granules, and C_i-granules and D-granules. We also present the efficient algorithms for constructing these deprived mappings based on the basic association mapping according to Theorem 6 and Theorem 7.

Algorithm 3 describes the procedure for calculating the derived association mapping, Γ_{ij}, which describes associations between C_i-granules and C_j-granules. In step 1, an empty is assigned to the set of C_i-granules as its initial value. For each C-granule, g (step 2), the algorithm splits it into two smaller granules: a C_i-granule, g_i, and a C_j-granule, g_j, in step 3. It then calculates the link-strength between the two smaller granules (step 4 to step 6), and updates the set of C_i-granules and the value of the association mapping (Γ_{ij}) (step 7 to step 12).

Figure 5.3 shows an example of a derived association mapping, Γ_{ij}, where the input parameters U_C and Γ_{cd} are described in Figure 5.1; the outputs of Algorithm 1; and the definitions of $cg_{i,1}$, $cg_{i,2}$, $cg_{j,1}$, $cg_{j,2}$ and $cg_{j,3}$ can be found

input : U_C, Γ_{cd}, C_i, C_j, where $C_i \cap C_j = \emptyset$ and $C_i \cup C_j = C$.
output: Γ_{ij}, a derived association mapping.

```
1  let U_i = ∅ ;
2  foreach g ∈ U_C do
3  |  let (g_i ∧ g_j) = g, s_1 = 0 ;
4  |  foreach (dg, s) ∈ Γ_cd(g) do
5  |  |  let s_1 = s_1 + s ;
6  |  end
7  |  if g_i ∈ U_i then
8  |  |  let Γ_ij(g_i) = Γ_ij(g_i) ∪ {(g_j, s_1)};
9  |  end
10 |  else
11 |  |  let Γ_ij(g_i) = {(g_j, s_1)}, U_i = U_i ∪ {g_i} ;
12 |  end
13 end
```

Algorithm 3: Calculating Γ_{ij}.

in Tables 3.7 and 3.8.

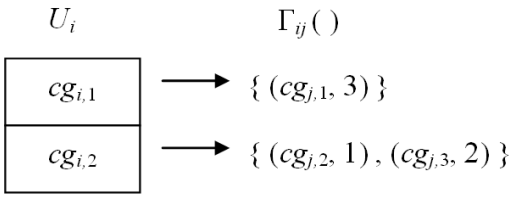


Figure 5.3: A derived mapping Γ_{ij}.

Figure 5.4 displays the process to build the C_i tier and mapping Γ_{ij}. This process begins by calculating the support for the granule, cg_1, which is three in the example. It then splits the C-granule, cg_1, into the granules of $cg_{i,1}$ and $cg_{j,1}$, and generates the mapping $\Gamma_{i,j}(cg_{i,1})$. Since the collection of C_i and $\Gamma_{i,j}$ is empty, the algorithm inserts the granules and mapping directly into these collections. After the insertion, the structure contains the granule $cg_{i,1}$ and mapping $\Gamma_{i,j}(cg_{i,1})$, where the count of $cg_{i,1}$ is three and $\Gamma_{i,j}(c_{i,1}) = \{(c_{j,1}, 3)\}$.

The process for the C-granule, cg_2, is similar to cg_1. The C_i granule, $cg_{i,2}$; C_j granule, $cg_{j,2}$; and the mapping, $\Gamma_{ij}(cg_2)$, are generated from cg_2. The count

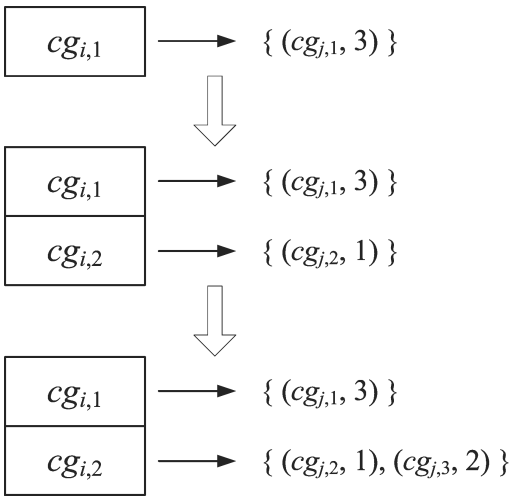


Figure 5.4: Example process of derived generation mappings of Γ_{ij}

for both generated granules and the link-strength of the generated mapping are both one. Because $cg_{i,2}$ and $\Gamma_{ij}(cg_2)$ are not found in the structure, the algorithm directly inserts these elements into the corresponding C_i and Γ_{ij} collection, respectively. After the insertion, the structure consists of granules $C_i = \{cg_{i,1}, cg_{i,2}\}$ and mappings $\Gamma_{ij} = \{\Gamma_{ij}(cg_{i,1}), \Gamma_{ij}(cg_{i,2})\}$.

Next, from the C-granule, cg_3, the algorithm generates a C_i-granule, which is the same as $cg_{i,2}$, and a Γ_{ij} mapping, $\Gamma_{ij}(cg_{i,2}) = \{(cg_{j,3},\ 2)\}$. Obviously, the algorithm can find the generated C_i-granule in the current structure, then starts the search for generated mapping in the collection of $\Gamma_{ij}(cg_{i,2})$. Finally, the algorithm is unable to find the mappings and thus it inserts new mappings into the current collection. When the process finishes, a C_i tier–including $cg_{i,1}$ and $cg_{i,2}$ with their mappings–$\Gamma_{ij}(cg_{i,1})$ and $\Gamma_{ij}(cg_{i,2})$–is constructed to the structure, as shown in the Figure 5.3.

The time complexity of using Algorithm 3 to construct Γ_{ij} is $O((n(C_i) +$

$n(C_j))N)$, based on the analysis for Algorithm 2. In Algorithm 3, the nested 'for' loop (steps 4 to 6) traverses the basic association mapping; therefore, it takes $O(n)$ in total, where $n = |(T/V^T)|$. The 'if-else' statement (steps 7 to 12) traverses all condition granules and checks g_i in U_i; therefore, it takes $O(n(C_i)n(C))$. Because $n(C) \leq n$, the time complexity of Algorithm3 is:

$$O(n(C_i) \cdot n) \tag{5.2}$$

that is much better than directly calculating Γ_{ij} from the set of transactions (note: if using Algorithm 1 to calculate Γ_{ij}, the time complexity is

$$O((n(C_i) + n(C_j)) \cdot N)$$

based on Eq. 5.1, where $N = |T|$, $n(C_i) = |U_i|$ and $n(C_j) = |U_j|$).

Algorithm 4 describes the process for calculating the derived association mapping, Γ_{id}, which describes associations between C_i-granules and D-granules. For each C_i-granule, g_i, it first assigns an empty to $\Gamma_{id}(g_i)$ in step 3. It also composes all paths (through Γ_{ij} and then Γ_{cd}) that start from g_i and end at the same decision granule (step 5 to step 8).

Figure 5.5 shows an example of a derived association mapping, Γ_{id}, where the input parameters, U_i, U_C, and Γ_{cd}, are described in Figures 5.3 and 5.1, respectively.

Figure 5.6 shows the process to create the derived mapping, Γ_{id}, for each C-

input : U_i, U_C, and Γ_{cd}.
output: Γ_{id}, a derived association mapping.

foreach $g_i \in U_i$ **do**
  /* Initialize the derived association mapping*/
  let $\Gamma_{id}(g_i) = \emptyset$;
end
foreach $g \in U_C$ **do**
  let $g_i \wedge g_j = g$;
  /* where g_i is a C_i-*granule*, and g_j is a C_j-*granule* */
  let $\Gamma_{id}(g_i) = \Gamma_{ij}(g_i) \oplus \Gamma_{cd}(g)$;
end

Algorithm 4: Calculating Γ_{id}.

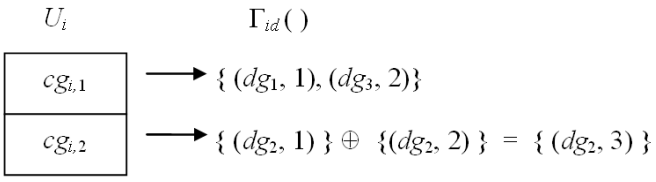


Figure 5.5: A derived mapping Γ_{id}.

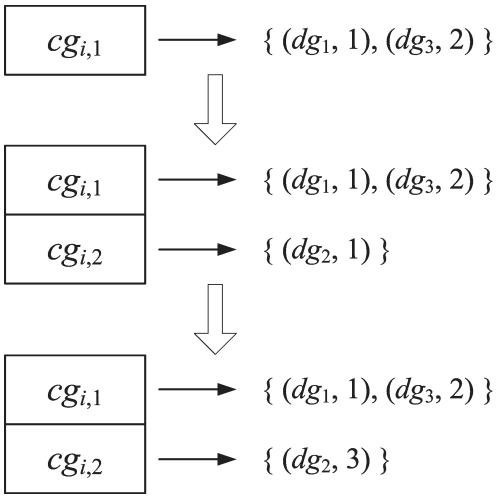


Figure 5.6: Example process of derived mapping generation of Γ_{id}

granule. The process begins immediately after the process of that C-granule to construct the C_i-granule and Γ_{ij} mappings. For example, the algorithm begins to build $\Gamma_{id}(cg_{i1})$ once the generation of cg_{i1} and $\Gamma_{ij}(cg_{i1})$ completes. The construction is conducted as the composition of $\Gamma_{id}(cg_{i1})$ and $\Gamma_{cd}(cg_1)$. Because $\Gamma_{cd}(cg_1)$ is

empty, the production is equal to $\Gamma_{cd}(cg_1) = (\{(dg_1, 1), (dg_3, 2)\}$. The algorithm then inserts this production and produces the $\Gamma_{id}(cg_{i1})$ collection as the first row of Figure 5.5. Next, the process of C-granule cg_2 and its mappings is the same as cg_1. When the process finishes, the generated $\Gamma_{id}(cg_{i2})$ is also equal to $\Gamma_{cd}(cg_2)$ $= (\{(dg_2, 1)\}$. For granule cg_3, since the generated Cg_i granule–which is cg_{i2}–is found in the collection of Cg_i, the algorithm then performs the composition between $\Gamma_{id}(cg_i2)$ and $\Gamma_{cd}(cg_3)$, which equals $\{(dg_2, 1)\} \oplus \{(dg_2, 2)\}$. The result is $\{(dg_2, 3)\}$ and is used to update the collection of $\Gamma_{id}(cg_{i2})$, such that, after the whole process accomplishes, the algorithm constructs the Γ_{id} mappings, as shown in Figure 5.5.

The time complexity of Algorithm 4 is determined by the second 'for' loop (step 5 to step 9). Because here each composition operation, $\oplus$, is equivalent to the basic operation (the comparison between granules). Therefore, the time complexity of this algorithm is:

$$O((n(C) \cdot n(C_i)) \tag{5.3}$$

This is much better than directly calculating Γ_{id} from the set of transactions (note: if use Algorithm 1 to calculate Γ_{id}, the time complexity is:

$$O((n(C_i) + n(D)) \cdot N)$$

based on Eq. 5.1, where $N = |T|$, $n(C_i) = |U_i|$ and $n(D) = |U_D|$).

5.3 Meaningless rule filtering

After constructing the multi-tier structure containing derived mappings, it is possible to prune the meaningless rules. The rules with a weaker strength than their general rules are marked and are not presented to the users.

Moreover, different conditions can be used to filter the meaningless rules according to structure configuration, such as the number of tiers and types of mappings generated. Therefore, several algorithms are designed to handle different situation accordingly.

input : U_C, U_i, Γ_{cd}, Γ_{id}.
output: Mapping collection without meaningless rule.

```
1  foreach g ∈ U_C do
2      let temp = 0;
3      foreach (dg, s) ∈ Γ_cd(g) do
4          let temp = temp + s;
5      end
6      foreach g_i ∈ U_i do
7          if g_i ⊂ g then
8              let temp_i = 0;
9              foreach (dg, s_i) ∈ Γ_id(g_i) do
10                 let temp_i = temp_i + s_i;
11             end
12             foreach (dg, s) ∈ Γ_cd(g) do
13                 foreach (dg, s_i) ∈ Γ_id(g_i) do
14                     if s ÷ temp ≤ s_i ÷ temp_i then
15                         Mark g → dg as meaningless
16                     end
17                 end
18             end
19         end
20     end
21 end
```

Algorithm 5: Meaningless rule filtering with 3-tier structure

Algorithm 5 describes the procedures of the meaningless rule filtering with a three-tier structure containing the derived mappings, Γ_{id}, which are used as

criteria. The algorithm traverses all the Γ_{CD} mappings, calculates the rule confidences and compares these confidences with the confidences of the corresponding general rules to filter the meaningless rules.

For each C-granule, g, the algorithm first calculates the count for g ($temp$) by traversing all mappings of Γ_{CD} (steps 3 to 5). Then it searches through the C_i-granules to find the g_i that satisfies that $g \succcurlyeq g_i$ (step 7). For every such g_i, the algorithm calculates the count for g_i ($temp_i$) by traversing the mappings of $\Gamma_{id}(g_i)$ (steps 9 to 11). Next, the algorithm traverses the $\Gamma_{id}(g_i)$ and $\Gamma_{CD}(g)$ mappings to find the match dg pairs (steps 12 and 13). For every match pair, the algorithm calculates the rule confidences of both mappings, and compares these confidences (step 14). According to the comparison result, the algorithm marks the rule $g \rightarrow dg$ as meaningless, or places the rule into the results set (step 15).

For the structures with more than three tiers and more than three types of mappings, Algorithm 5 can also undertake meaningless rule pruning using only one type of the mappings. For example, with a four-tier structure containing the mappings of $\Gamma_{(i,1)d}$ and $\Gamma_{(i,2)d}$, the general rules of one type of these mappings can be used in Algorithm 5 as the criteria by replacing the U_i with $U_{i,1}$ or $U_{i,2}$, g_i with $g_{i,1}$ or $g_{i,2}$ and Γ_{id} with $\Gamma_{(i,1)d}$ or $\Gamma_{(i,2)d}$ (between steps 6 and 10).

A variation from Algorithm 5 is to use the general rules of both the $\Gamma_{(i,1)d}$ and $\Gamma_{(i,2)d}$ mappings as criteria. The modified algorithm has the corresponding components to examine the rules against each type of mapping used as the filtering condition. Algorithm 6 is designed for using both mappings of $\Gamma_{(i,1)d}$ and

input : U_C, $U_{i,1}$, $U_{i,2}$, $\Gamma_{(i,1)d}$, $\Gamma_{(i,2)d}$.
output: Mapping collection without meaningless rule.

```
1  foreach g ∈ U_C do
2      let temp = 0;
3      foreach (dg, s) ∈ Γ_cd(g) do
4          let temp = temp + s;
5      end
6      foreach g_i ∈ U_{i,1} do
7          if g_{i,1} ⊂ g then
8              let temp_{i,1} = 0;
9              foreach (dg, s_{i,1}) ∈ Γ_id(g_{i,1}) do
10                 let temp_{i,1} = temp_{i,1} + s_{i,1};
11             end
12             foreach (dg, s) ∈ Γ_cd(g) do
13                 foreach (dg, s_i) ∈ Γ_{(i,1)d}(g_{i,1}) do
14                     if s ÷ temp ≤ s_{i,1} ÷ temp_{i,1} then
15                         Mark g → dg as meaningless
16                     end
17                 end
18             end
19         end
20     end
21     foreach g_{i,2} ∈ U_{i,2} do
22         if g_{i,2} ⊂ g then
23             let temp_{i,2} = 0;
24             foreach (dg, s_{i,2}) ∈ Γ_{(i,2)d}(g_{i,2}) do
25                 let temp_{i,2} = temp_{i,2} + s_{i,2};
26             end
27             foreach (dg, s) ∈ Γ_cd(g) do
28                 foreach (dg, s_{i,2}) ∈ Γ_{(i,2)d}(g_{i,2}) do
29                     if s ÷ temp ≤ s_{i,2} ÷ temp_{i,2} then
30                         Mark g → dg as meaningless
31                     end
32                 end
33             end
34         end
35     end
36 end
```

Algorithm 6: Meaningless rule filtering with 4-tier structure

$\Gamma_{(i,2)d}$ as the condition. In Algorithm 6, there is the corresponding component for filtering meaningless rule using $\Gamma_{(i,1)d}$ (step 6 to step 20), and another component for using $\Gamma_{(i,2)d}$ (step 21 to step 35) as criteria.

5.4 Estimated support calculation

With the multi-tier structure of granules and mappings, it is possible to calculate the estimated support for given patterns. As discussed in Chapter 4, there are several different cases in the calculation of estimated support. Therefore, several algorithms are developed for the corresponding cases.

The first algorithm for estimated support calculation is for the two-tier structure containing Γ_{CD} mappings, as presented as Algorithm 7. First, the algorithm splits the given pattern α into α_1 and α_2, such that $\alpha_1 \wedge \alpha_2 = \alpha$, according to the definition of C-granule and D-granule (step 7). Next, the algorithm selects the corresponding calculation method according to the status of α_1 and α_2 (steps 8 and 13). If α_2 is empty, it means that α is only derived from C-granules.

Thus, the algorithm traverse the granules of U_C to find the granule, g, such that $\alpha \subseteq g$, and sums the count of every g to get the total occurrence of α (steps 9 to 11). Similarly, steps 14 to 16 are for the case in which α_1 is empty. For the case in which none of α_1 or α_2 is empty, the algorithm first goes through the granules in U_C to find granule g containing α_1 (step 20), then traverses the mappings of $\Gamma_{CD}(g)$ to get the dg pairs where $\alpha_2 \subseteq dg$ (step 21). The algorithm adds up the strengths, s, of these dg pairs to get the total occurrence of α (steps

22 and 23). Finally, the estimated support, S'_α, is calculated by dividing the total count of α by the total count of N (step 29).

```
   input  : α, U_C, U_D and Γ_cd.
   output: S'_α, estimated support of α.
 1 let N = 0;
 2 foreach g ∈ U_C do
 3 |   foreach (dg, s) ∈ Γ_cd(g) do
 4 |   |   let N = N + s;
 5 |   end
 6 end
 7 Let α = α_1 ∧ α_2;
 8 if α_1 = ∅ then
 9 |   foreach g ∈ U_C do
10 |   |   S'_α = S'_α + count of g
11 |   end
12 end
13 else if α_2 = ∅ then
14 |   foreach g ∈ U_D do
15 |   |   S'_α = S'_α + count of g
16 |   end
17 end
18 else
19 |   foreach g ∈ U_C do
20 |   |   if α_1 ⊆ g then
21 |   |   |   foreach ((dg, s) ∈ Γ_cd do
22 |   |   |   |   if α_2 ⊆ dg then
23 |   |   |   |   |   S'_α = S'_α + s
24 |   |   |   |   end
25 |   |   |   end
26 |   |   end
27 |   end
28 end
29 S'_α = S'_α ÷ N;
```

Algorithm 7: Estimated support calculation with two tier structure

As discussed before, for a multi-tier structure with more than three tiers, there are more possible derivation cases for a given pattern. For example, with a three-tier structure, it is possible for a given pattern to be derived from the granules of one, two or three tiers. Thus, several algorithms are designed to calculate the estimated support for each corresponding case. Algorithms 8, 9, and 10 are for

the estimated support calculation based on a three-tier structure. For all the algorithms, the given pattern α is first divided into three smaller patterns–α_1, α_2 and α_3–according to the granule definitions. Then, based on the situation of these patterns, the corresponding algorithm is selected to conduct the calculation.

Algorithm 8 presents the calculation process for the cases (case 1) where α is derived from the granules of only one tier. According to the tier of granule which the pattern is derived, the algorithm traverses the granule in the corresponding tier and accumulates the counts of all granules containing the patterns. That is, if $\alpha_2 = \emptyset$ and $\alpha_3 = \emptyset$, the algorithm inspects each g in U_i and adds the count for each g containing α (step 8 to step 12) to total count of α. In addition, g_j and U_j are used if $\alpha_1 = \emptyset$ and $\alpha_3 = \emptyset$ (step 13 to step 17), while dg and U_D are used if $\alpha_1 = \emptyset$ and $\alpha_2 = \emptyset$ (step 18 to step 22).

Algorithm 9 describes the process for calculating the estimated support for the cases (case 2) where one of α_1, α_2 and α_3 is empty. The algorithm switches to the corresponding handler component according which sub-pattern is empty. If α_3 is empty, the algorithm searches through each g_i in U_i to find the C_i-granules containing α_1 (step 9 and step 10). Once such a g_i is found, then the algorithm turns to inspect the mappings of $\Gamma_{ij}(g_i)$ (step 11). If the algorithm discovers the g_j pair, such that $\alpha_2 \subset g_j$, it adds the link-strength to the count of α (step 23 and step 24). After the algorithm traverses all g_i, it calculates the estimated support at step 41. Similarly, the algorithm searches g_i in U_i and the mappings in $\Gamma_{id}(g_i)$ for the cases where α_2 is empty (step 19 to step 29), while it traverses g_j in U_j

input : α, U_i, U_j, U_D.
output: S'_α, estimated support of α.

```
1  let N = 0;
2  foreach g ∈ U_i do
3  |   foreach (dg, s) ∈ Γ_id(g) do
4  |   |   let N = N + s;
5  |   end
6  end
7  Let α = α_1 ∧ α_2 ∧ α_3;
8  if α_2 = ∅ and α_3 = ∅ then
9  |   foreach g ∈ U_i do
10 |   |   S'_α = S'_α + count of g
11 |   end
12 end
13 else if α_1 = ∅ and α_3 = ∅ then
14 |   foreach g ∈ U_j do
15 |   |   S'_α = S'_α + count of g
16 |   end
17 end
18 else if α_1 = ∅ and α_2 = ∅ then
19 |   foreach g ∈ U_D do
20 |   |   S'_α = S'_α + count of g
21 |   end
22 end
23 S'_α = S'_α ÷ N;
```

Algorithm 8: Estimated support calculation with 3-tier structure, case 1

and the mappings in $\Gamma_{jd}(g_j)$ if α_3 is empty (step 30 to step 40).

For cases in which none of α_1, α_2 or α_3 is empty (case 3), there are two options for estimated support calculation. As discussed in Chapter 4, one option is precise estimated support calculation, which can be performed through a two-tier structure. Therefore, Algorithm 7 can be used to describe the process of precise calculation in this situation. The second option is an approximate calculation. This calculation is computed through the minimum link-strength of the association mappings between the granules containing the sub-patterns of α. Algorithm 10 presents the process of this calculation. It starts from traversing the granule g_i in U_i to search for the C_i-granules containing α_1 (steps 2 and 3). When the algorithm finds such a g_i, it turns to search the mappings of $\Gamma_{ij}(g_i)$ to find the g_j pairs that contain α_2 (steps 4 and 5). For each matching pair, the algorithm traverses the mappings of $\Gamma_{jd}(g_j)$ for the matching pairs where dg contains α_3 (steps 6 and 7). For such a dg of each match mapping, the algorithm then traverses the mappings of $\Gamma_{id}(g_i)$ to find the matching dg pair containing the same dg (steps 8 and 9). If such a mapping is found, the algorithm selects the minimum link-strength from the dg pairs of (dg, s_1), (g_j, s_2) and (dg, s_3), and adds the corresponding link-strength to the count of α (step 10). Finally, the estimated support is computed in step 28.

```
input  : α, U_i, U_j, U_D, Γ_id, Γ_ij, Γ_jd.
output: S'_α, estimated support of α.
1  let N = 0;
2  foreach g ∈ U_i do
3  |  foreach (dg, s) ∈ Γ_id(g) do
4  |  |  let N = N + s;
5  |  end
6  end
7  Let α = α_1 ∧ α_2 ∧ α_3;
8  if α_3 = ∅ then
9  |  foreach g_i ∈ U_i do
10 |  |  if α_1 ⊆ g_i then
11 |  |  |  foreach ((g_j, s) ∈ Γ_ij(g_i)) do
12 |  |  |  |  if α_2 ⊆ g_j then
13 |  |  |  |  |  S'_α = S'_α + s
14 |  |  |  |  end
15 |  |  |  end
16 |  |  end
17 |  end
18 end
19 if α_2 = ∅ then
20 |  foreach g_i ∈ U_i do
21 |  |  if α_1 ⊆ g_i then
22 |  |  |  foreach ((dg, s) ∈ Γ_id(g_i)) do
23 |  |  |  |  if α_3 ⊆ dg then
24 |  |  |  |  |  S'_α = S'_α + s
25 |  |  |  |  end
26 |  |  |  end
27 |  |  end
28 |  end
29 end
30 if α_1 = ∅ then
31 |  foreach g_j ∈ U_j do
32 |  |  if α_2 ⊆ g_j then
33 |  |  |  foreach ((dg, s) ∈ Γ_jd(g_j)) do
34 |  |  |  |  if α_3 ⊆ dg then
35 |  |  |  |  |  S'_α = S'_α + s
36 |  |  |  |  end
37 |  |  |  end
38 |  |  end
39 |  end
40 end
41 S'_α = S'_α ÷ N;
```

Algorithm 9: Estimated support calculation with 3-tier structure, case 2

```
input  : α, U_i, U_j, U_D, Γ_id, Γ_ij, Γ_jd.
output: S'_α, estimated support of α.
1  let N = 0;
2  foreach g_i ∈ U_i do
3      if α_1 ⊆ g_i then
4          foreach (g_j, s_2) ∈ Γ_ij(g_i) do
5              if α_2 ⊆ g_i then
6                  foreach (dg, s_3) ∈ Γ_jd(g_j) do
7                      if α_3 ⊆ dg then
8                          foreach (dg, s_1) ∈ Γ_id(g_i) do
9                              if α_3 ⊆ dg then
10                                 S'_α = S'_α + Min(s_1, s_2, s_3);
11                                 if Min(s_1, s_2, s_3)=s_1 then
12                                     N = N + count of g_i
13                                 end
14                                 if Min(s_1, s_2, s_3)=s_2 then
15                                     N = N + count of g_j
16                                 end
17                                 if Min(s_1, s_2, s_3)=s_3 then
18                                     N = N + count of dg
19                                 end
20                             end
21                         end
22                     end
23                 end
24             end
25         end
26     end
27 end
28 S'_α = S'_α ÷ N;
```

Algorithm 10: Estimated support calculation with 3-tier structure, case 3

5.5 Summary

This chapter has presented a series of algorithms that implement the proposed approach. These algorithms include the construction algorithm for the basic and derived association mappings, the meaningless rule filtering algorithms and the algorithm for calculating the estimated supports.

The algorithm to construct the basic association mappings traverses all the transactions of the information table and generates the condition and decision granule with the basic association mappings between these granules. Following this, the algorithm for the derived association mapping construction goes through the condition granules to generate the smaller condition granules and construct the association mappings between these granules, as well as the mappings between the condition and decision granules in the upper tier. When the mapping generation completes, a multi-tier structure of granules is built.

The meaningless rule filtering algorithm then uses this multi-tier structure to identify the meaningless rules. This algorithm traverses all the decision rules in the two-tier structure and, for each decision rule, searches all the corresponding general rules in the multi-tier structure to compare the confidence. Moreover, according to the number of tiers in the structure, the corresponding algorithm is designed to use more than one kind of general rule to filter the meaningless rule.

For support estimation, a group of algorithms is developed to conduct the calculation for each case of the derivative of the given pattern. With a two-tier structure, the algorithm traverses the granules and uses the granule support

or link-strength to calculate the estimated support accordingly. With a three-tier structure, three algorithms are designed to handle each case of the deriving of the given pattern. For case 1, where the pattern is derived using only one tier's granules, the algorithm searches the granules and uses granule support to calculate the estimated pattern support. For case 2, the algorithm uses the link-strength of the corresponding mappings to conduct the calculation. Finally, for case 3, the algorithm searches the corresponding mappings containing the given pattern, and selects the minimum one to undertake the estimated support calculation.

For algorithm complexity, the complexity of the construction algorithm for basic mappings is determined by the number of conditions and decision granules. These numbers are less than the number of granules and much less than the number of total transactions, while the complexity for retrieving decision rules is decided by the number of granules. This performance is better than using decision tables. For the algorithms to construct the two kinds of derived mappings, the complexity depends on the number of granules in the tier to conduct the driving of the corresponding mappings. This performance is better than performing the same task directly using the transactions.

Chapter 6

Evaluation

To evaluate the performance of the multi-tier granule mining solution, a series of experiments–including comparison with the baseline models–were conducted on practical data sets. Two data sets–from a Foodmart 2005 data warehouse and from network traffic data–were used as the test bed. Corresponding measures were used for the space complexity, time complexity, accuracy of support estimation and scalability. Some examples of meaningless rules obtained from the test data were also used to illustrate the effect of the meaningless rules. Finally, this chapter presents the experiment results and a discussion based on the results.

6.1 Experiment purpose

Based on the research question and research task, we had the following reasons to conduct the experiment in our evaluation. First, the number of granules was tested to examine the performance of the proposed model in order to overcome the problem of the overwhelmingly large volume of patterns. Moreover, to check the performance in reducing number of rules, number of meaningless rules filtered is also tested with different configurations.

Second, to display how the proposed model uses the structural information, several tests were conducted using different granule definitions, which were defined according to the structural information of the data warehouse. Multi-tier structures in these tests were then constructed according to the hierarchy structure of the selected dimensions, which were used to define granules. Further, for meaningless rules, a test was performed to present some examples of meaningless rules and explain the meanings of these rules using the structural information of the testing data.

Finally, tests were also conducted to examine the accuracy of the support estimation through the proposed method. These tests also reflect the completeness of the granules and multi-tier structures. In particular, some tests were used to demonstrate that support estimation through granules can achieve a zero error.

6.2 Baseline models

There were two baseline models used for the evaluation. The first was the decision table model. Using the decision table as a baseline model is useful because the decision table is an approach based on rough set theory, which can be used for association rule mining. Since the decision table method creates some issues in association mining, as discussed previously, and because our approach aims to overcome these issues, the decision table was used as the baseline model in this evaluation. In our experiment, the attributes of the test data were divided into two groups: condition and decision attributes. Based on this division, a decision table was built. This decision table contained a set of decision rules that were used to calculate the estimated supports of given patterns. Further, the number of decision granules was also used to compare with the number of granules in the multi-tier structure.

The second baseline model was the profile-based pattern summarisation model. This model provides a method to reduce pattern numbers, and a method to restore patterns. Similarly, the granules and multi-tier structure are a kind of compressed representation of patterns, and pattern support can also be calculated from granules. Therefore, the profiled-based method was used as the baseline model to compare the performance of the two methods. As described in the previous chapter, this approach summarises the closed pattern into a certain number of profiles, then the estimated support is calculated through the profiles using Eq. 4.1. Specifically, the closed patterns are generated from the information table

using the Apriori algorithm in the experiment. Each attribute in the information table is treated as an item. Then, the frequent one-item patterns are generated from the information table and other frequent patterns are generated using the Apriori algorithm. All the counts and supports of the patterns are also obtained from the information table.

The closed patterns are summarised into profiles through clustering techniques. There are two methods used in the original approach: hierarchical agglomerative clustering and K-means clustering. Since K-means clustering has less computation cost, it was employed in this experiment. There are three steps in the clustering process. First, K patterns are randomly selected as the cluster centre. Second, these patterns are reassigned to the clusters according to the KL–divergence criterion. Third, the profiles of the newly formed clusters are updated. The second and third steps are repeated until there is only small change in all the profiles when the clusters are updated. However, the repeated calculation of KL-divergence requires high computational consumption. Therefore, in this experiment, a simplified summarisation without the repeated assignment was used–every pattern was assigned only once in the whole clustering process. That is, one closed pattern could only be assigned to one profile and could not be reassigned to another profile for the summarisation.

6.3 Measures

The experiments examined the performance of the proposed approach from several aspects, including the space complexity, time complexity, accuracy of support estimation and scalability. Space complexity is used to examine the effectiveness of reducing the number of patterns and association rules. Time complexity reflects the efficiency of our methods in generating granules and constructing multi-tier structures. The accuracy of support estimation is the measure used to show how close the support calculated through our approach is to the original one. Finally, the scalability shows the performance consistency of the proposed model.

Space complexity is examined by checking the numbers of granules and patterns. In the multi-tier structure, the numbers of the granules in different tiers are the main value to demonstrate the space complexity of the granule mining approach. Moreover, multi-tier structures built under different granule definitions can illustrate the performance of space complexity in a variety of conditions. The space complexity for the pattern-based method is represented as the number of patterns obtained under different support settings, or the different number of transactions used for pattern mining.

Time complexity is used to measure the efficiency of the approach. For the granule mining approach, it is the time consumed to construct the multi-tier structure from the decision table, and the time required to construct more tiers from the existing structure. For the pattern-based methods, the time complexity

is the time used to obtain all the frequent patterns and closed patterns, and the time used to summarise these patterns into profiles.

For the accuracy of the estimated support, the restoration error rate is used as the performance measure. The restoration error rate reflects the correctness of the calculation of the estimated support for given patterns. The estimated supports are calculated by pattern profiles for the baseline model. For our approach, the estimated supports were calculated through granules and mappings. The restoration error rate is denoted as J and defined as follows:

$$J = \frac{1}{|T|} \sum_{\alpha_k \in T} \frac{|s(\alpha_k) - \hat{s}(\alpha_k)|}{s(\alpha_k)} \tag{6.1}$$

In the above equation, $T = \{\alpha_1, \alpha_2, \ldots, \alpha_l\}$ is the set of given patterns; $s(\alpha_k)$ is the real support of the pattern, α_k; while $\hat{s}(\alpha_k)$ is the estimated support calculated by the pattern profiles or granules. The restoration error measures the average relative error between the real support and estimated support. Specifically, the original pattern sets can be used as the testing pattern set so that the restoration error measures the difference between the real support and the estimated support. The smaller the error rate, the closer the estimated support to the actual support. If the restoration error is zero, the estimated support equals the actual support.

For scalability, the numbers of granules and meaningless rules were used as the measure. These numbers were recorded against the numbers of the transactions used to generate the granules and filter the meaningless rules. The changes of

these numbers demonstrate the scalability of the proposed approach.

6.4 Experiments on the Foodmart 2005 data

For the Foodmart 2005 data, a set of experiments were conducted to examine the space and time complexity. In accordance with the data schema and hierarchy, a series of multi-tier structures were constructed with different granule definitions. The tests for meaningless rules filtering were also conducted on these multi-tier structures. Finally, for the accuracy of estimated support, an experiment was performed to calculate the estimated support through the multi-tier structure, and to compare this with the estimated support calculated by profiles with different profile numbers.

6.4.1 Details of the Foodmart 2005 data

The Foodmart 2005 dataset used in the experiment was a sample data warehouse for the SQL Server that was provided with the book Microsoft SQL Server 2008 Analysis Services Unleashed. The sample data warehouse can be downloaded from: *http://www.e-tservice.com/*. It contains two databases: Foodmart 2005 SQL Database and Foodmart 2005 OLAP. The data used in the experiment were customer sales data from the Foodmart 2005 OLAP database. This data set was used so we could employ the structural information from the data set to define granules of different sizes. Moreover, the hierarchy information could also be used to describe the multi-tier structure of granules.

The Foodmart 2005 OLAP database consists of a set of cubes and a set of dimensions. Within this, there are four cubes: budget, HR, sales and employees, and warehouse and sales. The set of dimensions has 11 dimensions: account, category, currency, customer, department, employee, product, promotion, store, time and warehouse. Each data cube contains several measure groups. For example, the 'warehouse and sales' cube, which was used in the experiment, contains four measure groups: rate, sales, warehouse and warehouse inventory. Each data cube also has a subset of all the dimensions that comprise the facts of the measures in that cube. For the 'warehouse and sales' cube, it has seven dimensions: currency, customer, product, promotion, store, time and warehouse.

The measure for unit sales was selected as the actual transaction for the experiment. Figure 6.1 depicts the structure of the unit sales measure, which uses the product, fate, store, promotion, customer and currency dimensions. The data used in the experiments actually only include the product, time and customer dimensions. All these dimensions have a hierarchy containing multiple levels. For example, the 'time' dimension has the levels of year, month and date, while the 'customer' dimension has four levels–country, state province, city and customer. The 'product' dimension consists of seven levels: product family, product department, product category, product subcategory, brand and product level. The unit sales measure uses the attributes in the levels of date, customer and product from the corresponding dimensions to generate the measure values. Each row of the unit sales data is a transaction of a customer's purchase of the products in one

day.

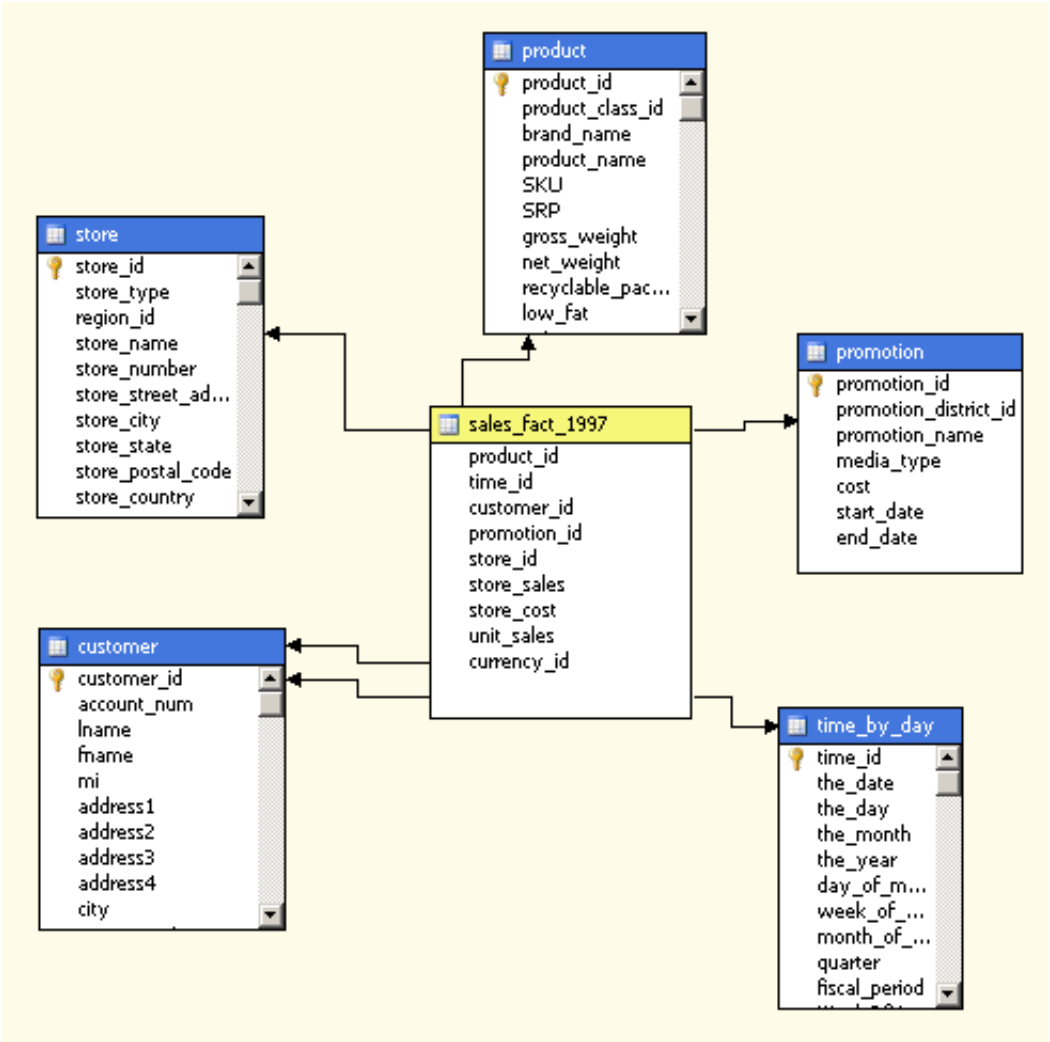


Figure 6.1: Data structure of *Unit sales* measure

The data used in the experiment had some variation from the 'unit sales' data in terms of the 'product' dimension. The data used in the experiment used the attributes of the 'product department' level, instead of the attributes from the 'product' level of the 'Product' dimension. That is, the data used in the experiment were actually the roll-up data of the unit sales measure. The reason for using the 'product department' was that there were too many attributes in the

'product' level, and it was difficult to generate very long frequent patterns. Using the attributes of the 'product department' level, the values of the attributes of 'product' level under the same product department were summed to the value of the corresponding attribute in the 'product department' level. Figure 6.2 shows part of the unit sale measure values. In Figure 6.3, the highlighted row is the sales unit shown in the 'product' category level for the 'beverage' product department. The values for the categories are summed up to 60 of 'beverage', as shown in Figure 6.2.

	Product Family ▾ Product Department Product Category							
	⊟ Drink						⊞ Food	⊞ Non-Consumable
	⊞ Beverages		⊞ Alcoholic Beverages		⊞ Dairy	Total		
Customer ▾ Date ▾	Unit Sales		Unit Sales		Unit Sales	Unit Sales	Unit Sales	Unit Sales
⊞ A. Catherine Binkley	8					8	111	23
⊞ A. Joyce Jarvis					5	5	43	2
⊞ Aaron Carabellos							2	17
⊞ Aaron Conklin					2	2	14	5
⊞ Aaron Cope							1	2
⊞ Aaron Haddix							5	
⊞ Aaron Keane							15	12
⊞ Aaron Lemay	6				4	10	41	8
⊞ Aaron McDonnell	60		14		33	107	558	192
⊞ Aaron Quintan	12		4		3	19	83	21
⊞ Aaron Shehorn							17	
⊞ Aaron Story	14					14	173	50
⊞ Aaron Van Ness							5	
⊞ Aaron Whitteker	9					9	155	36

Figure 6.2: *Unit sales* shown in *Product deparment* level

	Product Family ▾ Product Department Product Category						
	⊟ Drink						
	⊟ Beverages						
	⊞ Carbonated Beverages		⊞ Drinks	⊞ Hot Beverages		⊞ Pure Juice Beverages	Total
Customer ▾ Date ▾	Unit Sales		Unit Sales	Unit Sales		Unit Sales	Unit Sales
⊞ A. Catherine Binkley				3		5	8
⊞ A. Joyce Jarvis							
⊞ Aaron Carabellos							
⊞ Aaron Conklin							
⊞ Aaron Cope							
⊞ Aaron Haddix							
⊞ Aaron Keane							
⊞ Aaron Lemay						6	6
⊞ Aaron McDonnell	11		18	19		12	60
⊞ Aaron Quintan						12	12
⊞ Aaron Shehorn							

Figure 6.3: *Unit sales* shown in *Product category* level

Figure 6.4 shows the attributes in the 'product department' level. Due to the

original database design, 'baking goods' and 'dairy' were duplicated in the 'drink' and 'food' families. Thus, these attributes in the 'food' family were removed so that 23 attributes in total were used to build the information table.

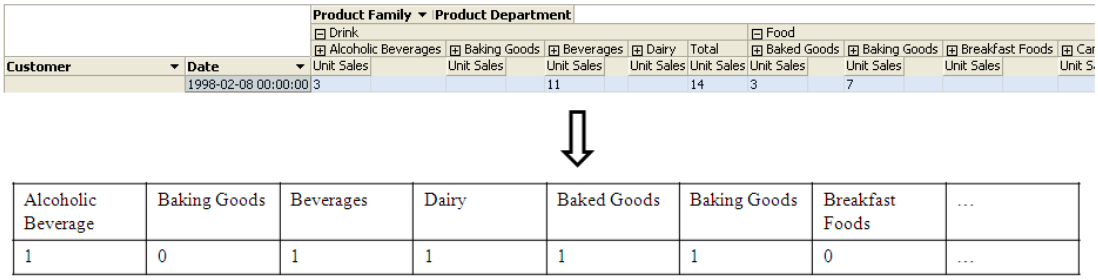
Figure 6.4: *Product* attributes in *Product department* level

In the experiment, these 23 attributes of the 'product department' were divided into different tiers to define the granules in that tiers according to the user constrains. Moreover, the division of these attributes can also follow the classification of the product family so that the granules in different tiers represent the corresponding 'product family'. This enabled the granule to carry

more structural meanings. In the experiment, because the 'food' family had 16 product departments, these product departments were divided into two groups to form the 'food 1' and 'food 2' families. This division was used to define the granules in a four-tier structure. These attributes were categorised into four product families: drink (alcoholic beverages, baking goods, beverages, dairy), non-consumables (carousel, checkout, health and hygiene, household, periodicals), food 1 (baked goods, breakfast foods, canned foods, canned products, deli, eggs, frozen foods) and food 2 (meat, packaged foods, produce, seafood, snack foods, snacks, starchy foods).

To build the decision table and the multi-tier structure of the granules, the transactions of the unit sales first needed to be transformed into the information table. The details of this transformation are as follows. If a customer purchases one or more products from the 'product department', the value of the attribute in the 'product department' level is set to one. Otherwise, the value is set to zero. Figure 6.5 presents an example of this transformation. This figure shows part of a transaction in which purchases occurred for all product departments under the 'drink' family, except 'baking goods'. Correspondingly, the value of 'baking goods' in the information table was set to zero, and the other attributes in the 'product department' level were set to one. This transformation then generates a transaction with the values of 1011..., as shown in the figure. Finally, all the transactions are transformed into a table with data rows containing the values of one and zero. The total number of transactions in the fact table is 53,700. All

or part (such as the top 100 sales or the sales of customers in one city) of the transactions are used according to the experiment configuration.



Alcoholic Beverage	Baking Goods	Beverages	Dairy	Baked Goods	Baking Goods	Breakfast Foods	...
1	0	1	1	1	1	0	...

Figure 6.5: Example of transformation of the transactions

6.4.2 Results

As described above, an information table was initially generated, then the decision table and multi-tier structures were constructed from this. The frequent patterns were also generated from this information table. Table 6.1 shows the attributes that define the granules in different tiers, and the relationship between the 'Product' dimension level and tiers. Tables 6.2, 6.3, 6.4 and 6.5 present the attribute details of the granule definitions for $C_{i,1}$, $C_{i,2}$, C_j and D tier, respectively. As shown in these tables, in the four-tier structure, each tier corresponds to an attribute in the 'product department' level, which includes a group of products. Table 6.6 shows the number of products in the granules of each tier and the granule numbers in that tier. This definition of the multi-tier structure was used in the second and third experiments.

The first experiment measured the space complexity. The test data used in this experiment included all 53,700 transactions from the unit sales measure.

Product Level	Tier Level			
All	C			D
Family	C_i		C_j	D
	Drink	Food1	Non-consumable	Food2
Department	$C_{i,1}$	$C_{i,2}$	C_j	D
	Alcoholic Beverage, ... , Dairy	Baked Goods, ..., Frozen Foods	Carousel, ..., Periodicals	Meat, ..., Starchy Foods

Table 6.1: The attributes of tiers

Attribute in $C_{i,1}$	A_1	A_2	A_3	A_4
Products of Drink	Alcoholic Beverages	Baking Goods	Beverages	Dairy

Table 6.2: Product attributes in $C_{i,1}$ tier

From the information table built from these transactions, a decision table was created. A two-tier structure containing tiers C and D was also built separately from the information table. From the two-tier structure, a three-tier structure was then constructed, from which a four-tier structure was built. The three-tier structure was built by dividing the C tier into two smaller tiers, C_i and C_j.

Attribute in $C_{i,2}$	A_5	A_6	A_7
Products of Food1	Baked Goods	Breakfast Foods	Canned Foods

Attribute in $C_{i,2}$	A_8	A_9	A_{10}	A_{11}
Products of Food1	Canned Products	Deli	Eggs	Frozen Foods

Table 6.3: Product attributes in $C_{i,2}$ tier

Attribute in C_i	A_{12}	A_{13}	A_{14}	A_{15}	A_{16}
Products of Non-Consumable	Carousel	Checkout	Health and Hygiene	Household	Periodicals

Table 6.4: Product attributes in C_j tier

Attribute in D	A_{17}	A_{18}	A_{19}
Products of Food2	Meat	Packaged Foods	Produce

Attribute in D	A_{20}	A_{21}	A_{22}	A_{23}
Products of Food2	Seafood	Snack Foods	Snacks	Starchy Foods

Table 6.5: Product attributes in D tier

Tier	Attribute number	N_g
Decision table	23	8373
C	16	2429
D	7	54
C_i	11	513
C_j	5	28
$C_{i,1}$	4	8
$C_{i,2}$	7	112

Table 6.6: Granule number in a 4-tier structure

Following this, the C_i tier was further divided into tiers $C_{i,1}$ and $C_{i,2}$ to construct the four-tier structure. Finally, a four-tier structure consisting of tiers $C_{i,1}$, $C_{i,2}$, C_j and D was constructed. In this experiment, the decision table and multi-tier structure were constructed under different configurations to demonstrate the performance for the space complexity.

There were 16 tests conducted in this experiment. The purpose of running these tests was to examine the granule numbers in a tier when there were different numbers of attributes used to define the granules. These tests also display the difference between the numbers of large and small granules. First, the test configuration began by allocating the least number of attributes to all tiers, except one, such that the granule in this tier had the largest number of attributes, while the granules in other tiers had the smallest number. Second, the tier with the largest granules was switched to another tier so that the test could show the difference in

granule numbers when different tiers had the largest number. Next, the number of attributes of the smaller granules was increased to show the change of granule numbers in all tiers. Finally, the test configuration was set to let all the tiers contain larger granules, except one tier that contained the smallest granules.

Table 6.7 shows the attribute numbers of granules in each tier under different configurations. The setting began by putting only one attribute in all tiers, except one tier with the most number of attributes in it. The number of attributes was then increased to two, four and six for different tests. For example, in test five, 21 attributes were allocated to tier C, and two were allocated to tier D. The attributes in tier C were then divided into 19 attributes that were used for tier C_i, and two that were used for tier C_j. Finally, the attributes for C_i were split into tiers $C_{i,1}$ and $C_{i,2}$, which had 17 and two attributes, respectively. Specifically, when there were six attributes in all other tiers, the remaining tier had the least attributes. For example, in test 23, there were five attributes in tier D, while all condition tiers had six attributes.

Table 6.8 shows the numbers of granules in the tiers with different granule definitions. It also shows the number of granules in each tier when the larger granules were divided into smaller granules. The overall results show that the number of granules in one tier and total granules in the whole multi-tier structure declined with the reduction of attributes in the tier or there are more tiers in the multi-tier structure. The first column in Table 6.8 is the number of granules in the decision table. It implies that if all 23 attributes were used for the tier, there

	All	C	D	C_i	C_j	$C_{i,1}$	$C_{i,2}$
Test1	23	22	1	21	1	20	1
Test2	23	22	1	21	1	1	20
Test3	23	22	1	2	21	1	1
Test4	23	3	20	2	1	1	1
Test5	23	21	2	19	2	17	2
Test6	23	21	2	19	2	2	17
Test7	23	21	2	4	17	2	2
Test8	23	6	17	4	2	2	2
Test9	23	19	4	15	4	11	4
Test10	23	19	4	15	4	4	11
Test11	23	19	4	8	11	4	4
Test12	23	12	11	8	4	4	4
Test13	23	17	6	11	6	5	6
Test14	23	17	6	11	6	6	5
Test15	23	17	6	12	5	6	6
Test16	23	18	5	12	6	6	6

Table 6.7: Attribute numbers in tiers

would be 8,737 granules in the tier. Not all tiers with less than 23 attributes have more granules than this number. From the results in tests one, two and three, the number of granules in tier C was more than 1,000 less only with one attribute used for decision tier D to create a two-tier structure. When the attributes in one tier were reduced to around 10, the granules in the tier were reduced to hundreds. Finally, with only a few attributes in the granule, the number of granules was only 10 or less.

Figure 6.6 depicts the trends of total granule numbers in the multi-tier structure when the number of tiers increased. In most of the tests, the number of granules dropped largely, while the number of tiers increased; although, in some tests, the number did not reduce by much. This was because the tier being divided had few attributes so that the newly generated tiers had a similar number

	All	C	D	C_i	C_j	$C_{i,1}$	$C_{i,2}$
Test1	8737	7521	2	6222	2	4506	2
Test2	8737	7521	2	6222	2	2	5025
Test3	8737	7521	2	2	6179	2	1
Test4	8737	4	5622	2	2	2	1
Test5	8737	6222	4	4022	4	2814	2
Test6	8737	6222	4	4022	4	2	3143
Test7	8737	6222	4	8	3011	2	4
Test8	8737	32	2933	8	4	2	4
Test9	8737	4020	16	1909	8	513	16
Test10	8737	4020	16	1909	8	8	632
Test11	8737	4020	16	107	404	8	16
Test12	8737	643	424	107	16	8	16
Test13	8737	2814	31	513	48	16	61
Test14	8737	2814	31	513	48	32	32
Test15	8737	2814	31	643	30	32	53
Test16	8737	2814	31	643	30	32	53
Average	8737	4076	573	1678	612	498	566

Table 6.8: Granule number in tiers of different granule definition

of granules.

For comparisons, frequent patterns and closed patterns were also generated from the information table. The results are shown in Table 6.9. The information table generated using all transactions was used in this test. The information table contained all the information of the transactions in the dataset; thus, all possible patterns were generated from the transaction at first. That is, all patterns with a support of one occurrence were treated as frequent patterns. The number of patterns was very large, as shown in Table 6.9. The closed patterns generated from these frequent patterns were much less, but still more than 15,000. With the minimum support set to five, the number of frequent patterns reduced to about 12,000. However, the closed patterns did not reduce significantly. Until the minimum support was set to 50, the number of frequent patterns and closed

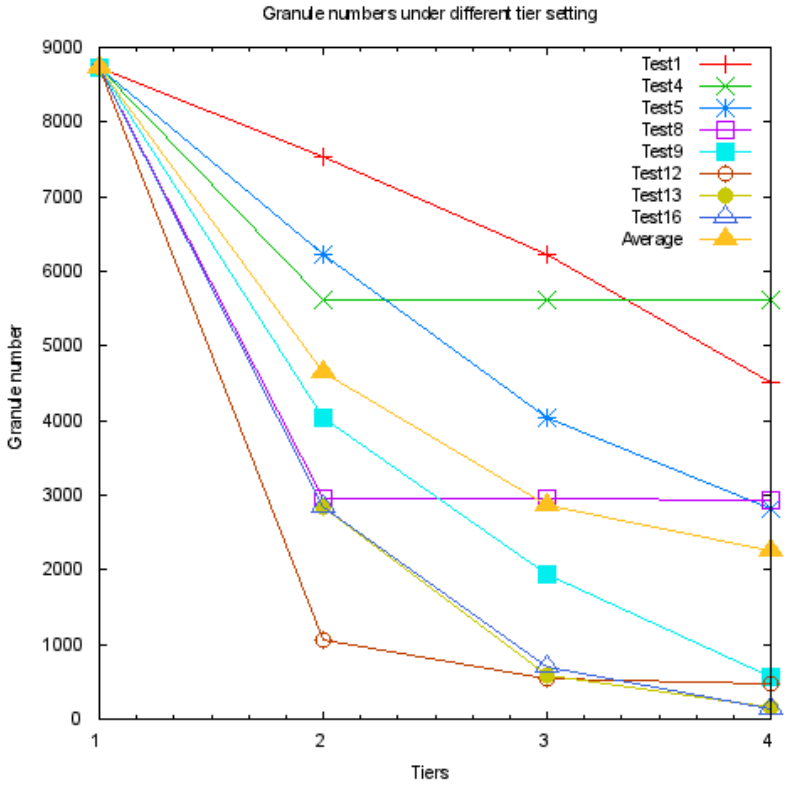


Figure 6.6: Granule numbers under different tier settings

patterns could be reduced to about 1,500.

Pattern	Min frequency	N
Frequent pattern	1	56707
Closed pattern	1	15859
Frequent pattern	5	12217
Closed pattern	5	10963
Frequent pattern	50	1486
Closed pattern	50	1486

Table 6.9: Frequent pattern number

In the second experiment, tests were conducted to measure the performance of the granule mining approach for time complexity, and compare the runtime with the decision table and pattern-based approach. The runtime of the granule mining approach is the time used to construct the multi-tier structure of granules.

The definition of the decision table and the granule of each tier are as shown in Table 6.1. The test began by building a two-tier structure from the information table. Following this, the time required to build the structure with a new tier constructed from the division of a larger granule tier was recorded, until a four-tier structure was constructed. For the pattern approach, the runtime is the time required to obtain frequent patterns under different minimum supports, and the time used to generate closed patterns from these frequent patterns.

Table 6.10 shows the results of the runtime tests. These results indicate that the time used by the granule mining approach was much less than that of the pattern-based approach. Only when the minimum support was set to a large number of occurrences, the time used to obtain the frequent patterns was close to the time used to construct the four-tier structure. Moreover, the results demonstrate that the time used to create new tiers from a smaller granule was less than that from larger granules. For example, the time used to derive the four-tier structure from the three-tier structure was 171 ms, while it was 2,593 ms to build the three-tier structure from the two-tier structure.

Granule		Pattern	
Multi-tier structure	Runtime(ms)	Minimum Support	Runtime(ms)
Decision table	19140	1	1.0778e+007
2-tier	6765	5	1.13072e+006
3 tier from 2 tier	2593	50	122672
4 tier from 3 tier	171		

Table 6.10: Runtime

The third experiment measured the restoration error rate, J, of the estimated support calculated by the granule approach, and compared the performance to

the profile-based pattern summarisation. The closed patterns generated from the whole information table with a minimum support of five occurrences were used as the input patterns for the estimated support calculation. There were 10,963 closed patterns in total. The actual support of every closed pattern was used to calculate the difference between the original and the estimated support to obtain the error rate of restoration for this pattern. The restoration error rate was the average difference of the whole set of closed patterns.

Since the number of profiles used for the summarisation can affect the restoration error rate, several tests were conducted with different numbers of profiles for pattern summarisation. As the number of closed patterns was 10,963, the number of profiles was accordingly set to 200, 500, 750 and 1,000. Setting the number of profiles to a larger number can reduce the J value because every profile covers fewer patterns; thus, the probability of an item in the master pattern is closer to the item's real support. However, the computation cost also increases. Moreover, the number of profiles should be kept as small as possible to reduce the number of patterns. Therefore, the number of profiles was not set to more than 1,000 in the tests.

Table 6.11 and Figure 6.7 show the comparison of the J values between the pattern- and granule-based approaches. The results of the pattern-based approach reflect that, when using small number of profiles–such as 200, 500 and 750–the restoration error rate is higher than when using granules. All these number of profiles are less than 10% of the total number of the given patterns or even

5%. When the number of profiles was set to 1,000, which was almost 10% of the total closed patterns, the J value of the profile summarisation approach was close to that of the granule mining approach. It should be noted that, when using a two-tier structure to calculate the estimated support, it is possible to achieve a zero restoration error rate. This result proves the discussion in Chapter 4 that the support estimated by the granules and mappings in a two-tier structure equals the pattern's real support. According to the properties of the profile-based pattern summarisation, if a much larger number of profiles are used, the J value can become smaller than that in the granule mining approach. However, the overall performance would be worse because the computation cost increases.

Profile		Decision table		Multi-tier of Granuls	
Profile number	J	Method	J	Method	J
200	0.931	Dicision rule	0	All mappings	0
500	0.924			Part of Mappings	0.822
750	0.891				
1000	0.864				

Table 6.11: Restoration error rate J

The granule mining approach has the special feature that it provides general rules with the multi-tier structure, and uses these general rules to prune meaningless rules. Table 6.12 lists the general rule numbers generated under different settings. As with the previous experiments examining the number of granules under different definitions, if less attributes were used to define the granules in the tiers from which the rules were generated, less general rules were generated. Except for the two-tier structure, there were no general rules for this structure. The number of rules remained at 8,737, regardless of the configuration. Figure 6.8

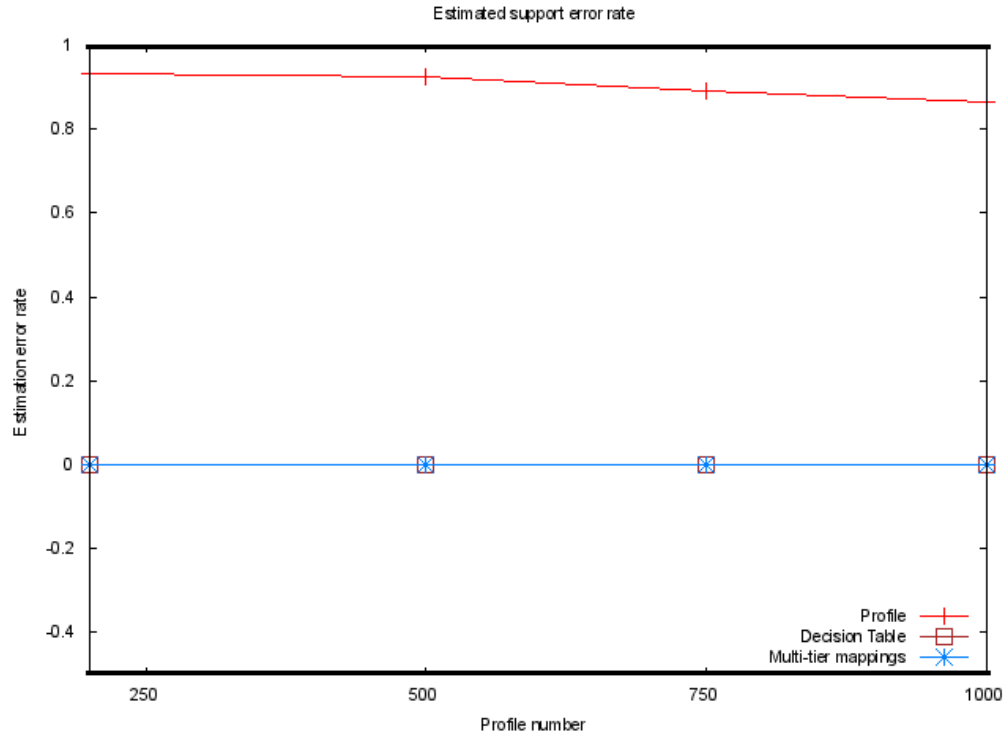


Figure 6.7: Estimated support error rate

displays the change of general rule numbers for the tests using different granule definitions.

With general rules, the meaningless rule can be removed from the result set presented to user. Table 6.13 presents the numbers of the meaningful and meaningless rules in the total 8,737 rules obtained under different definitions of granules for the tiers. The results demonstrate that the average number of meaningless rules was about 30% of the number of meaningful rules. Further, when only one type of general rule was used as the criteria, a fewer number of meaningless rules could be filtered. When two types of general rule–such as C_i and $C_{i,1}$–were used together as the criteria, more meaningless rules could be pruned. Figure 6.9 shows these changes in meaningless rule numbers under different pruning configurations.

General rule type	C to D	$C_i$to D	$C_{i,1}$to D
Test1	8737	7337	5442
Test2	8737	7337	4
Test3	8737	4	4
Test4	8737	6855	6855
Test5	8737	6002	4392
Test6	8737	6002	8
Test7	8737	32	8
Test8	8737	6241	3877
Test9	8737	5121	2036
Test10	8737	5121	109
Test11	8737	664	109
Test12	8737	4070	1425
Test13	8737	3026	335
Test14	8737	3026	500
Test15	8737	3332	500
Test16	8737	3332	500
Average	8737	4219	1632

Table 6.12: General rule numbers

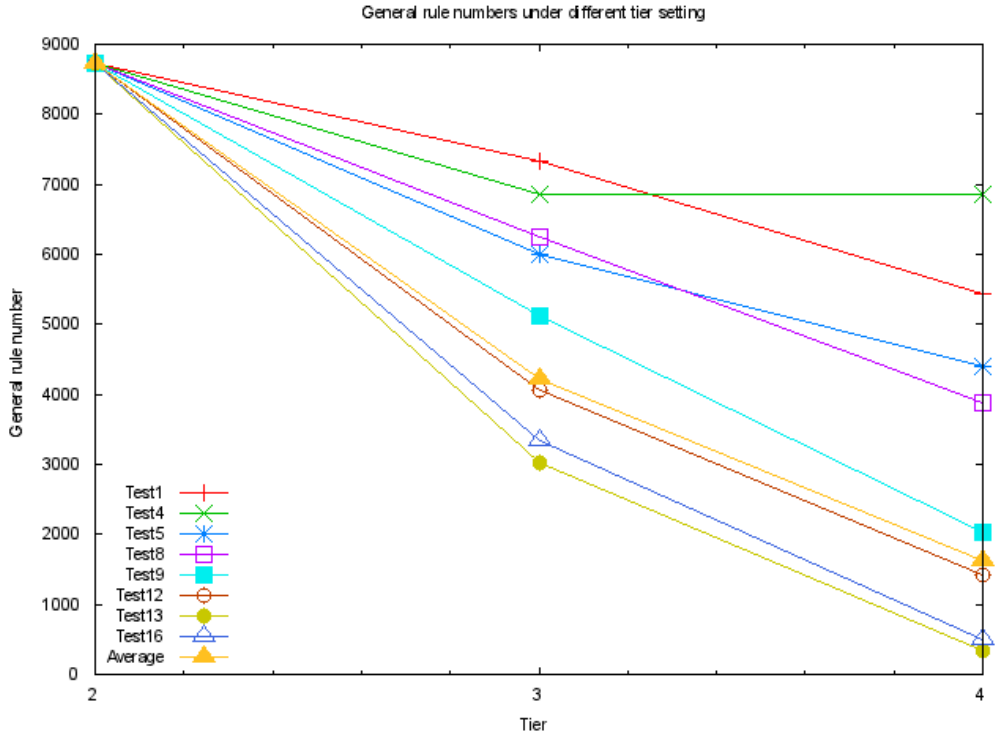


Figure 6.8: General rule numbers in different tier

Criteria	$C_i$to D		$C_{i,1}$to D	
Rule	Meaningful	Meaningless	Meaningful	Meaningless
Test1	7626	1111	7930	807
Test2	7521	1216	7930	807
Test3	7521	1216	7521	1216
Test4	6855	1882	6855	1882
Test5	7209	1708	7149	1588
Test6	6992	1745	7149	1588
Test7	6992	1745	7012	1725
Test8	6562	2175	6918	1819
Test9	6691	2046	6690	2047
Test10	6776	1961	6690	2047
Test11	6776	1961	6753	1984
Test12	6719	2018	6688	2049
Test13	6706	2031	6677	2060
Test14	6701	2036	6677	2060
Test15	6701	2036	6678	2059
Test16	6701	2036	6678	2059
Average	7046	1808	7000	1737

Criteria	$C_i$to D and $C_{i,1}$to D	
Rule	Meaningful	Meaningless
Test1	7443	1294
Test2	7272	1465
Test3	7521	1216
Test4	6885	1882
Test5	6555	2072
Test6	6505	2302
Test7	6948	1789
Test8	6085	2652
Test9	6125	2612
Test10	6064	2673
Test11	6586	2151
Test12	6229	2508
Test13	6232	2505
Test14	6231	2506
Test15	6212	2525
Test16	6212	2525
Average	6569	2167

Table 6.13: Meaningful and meaningless rule numbers

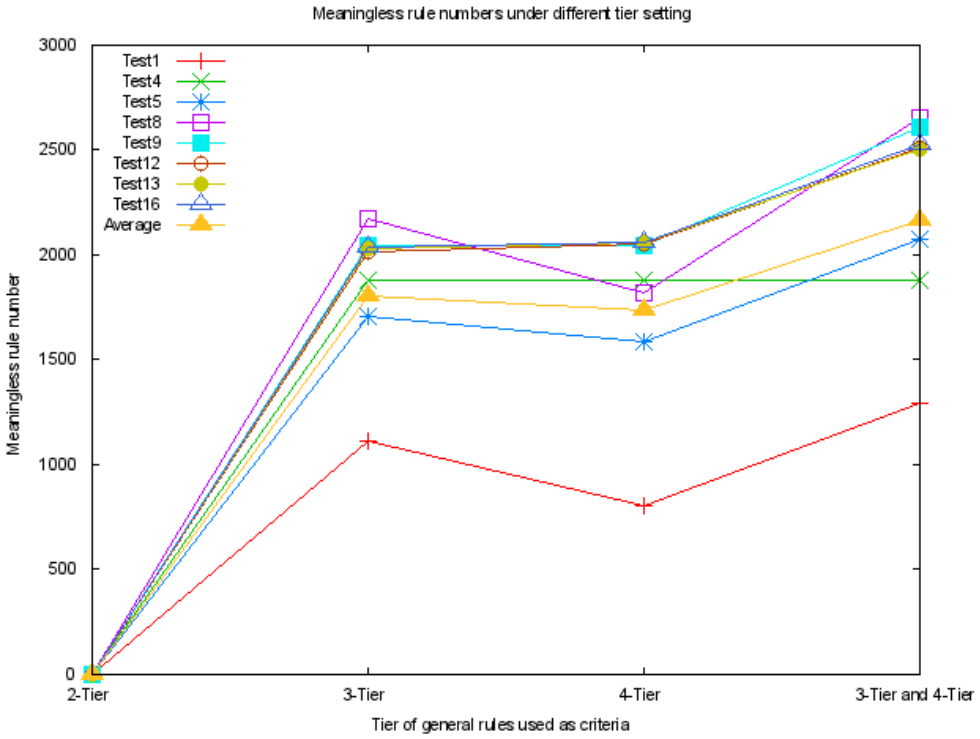


Figure 6.9: Meaningless rule numbers in different tier

Table 6.14 presents the construction runtime for the multi-tier structures under the 16 different configurations of the previous experiments. The table also shows the time consumed by meaningless rule pruning under these settings. Building the two-tier structure took the longest time to finish because it needed to read the data, which were large. The time used to construct the three-tier and four-tier structures was much less. The time was shorter when more tiers were built in the structure. In addition, the meaningless rule pruning was completed more efficiently when there were more tiers in the structure. These trends are illustrated in Figure 6.10.

runtime(MS)	Construction			Meaningless rule filtering	
Task	Database to 2-tier	2-tier to 3-tier	3-tier to 4-tier	in 3-tier	in 4-tier
Test1	33359	10828	5812	234	281
Test2	37453	10953	11328	218	1562
Test3	34297	16187	5687	5281	671
Test4	47969	12093	562	12093	1468
Test5	23062	4609	2234	203	203
Test6	22750	4546	3593	203	796
Test7	22734	3875	1375	1281	250
Test8	17078	1546	375	3015	531
Test9	12328	1296	296	171	140
Test10	12406	1296	375	171	171
Test11	12515	406	250	203	78
Test12	4140	281	218	375	125
Test13	7937	343	156	156	109
Test14	8687	359	140	171	93
Test15	7828	390	156	171	109
Test16	7718	390	140	156	109
Average	19516	4337	2043	1506	418

Table 6.14: Construction and meaningless rule pruning runtime

6.5 Experiments of the network traffic data

To further demonstrate the performance of the proposed approach, a second set of experiments was conducted on network traffic data that contained internet protocol (IP) addresses as attributes. This set of experiments has the following objectives. First, it demonstrates that the granule mining approach is a suitable tool for some scenarios, such as network traffic analysis, to which the traditional data mining approach cannot be applied directly. Second, it examines the scalability of the granule mining approach using a larger volume of data. Finally, it presents some examples to illustrate the effect of meaningless rules based on the natures of IP addresses.

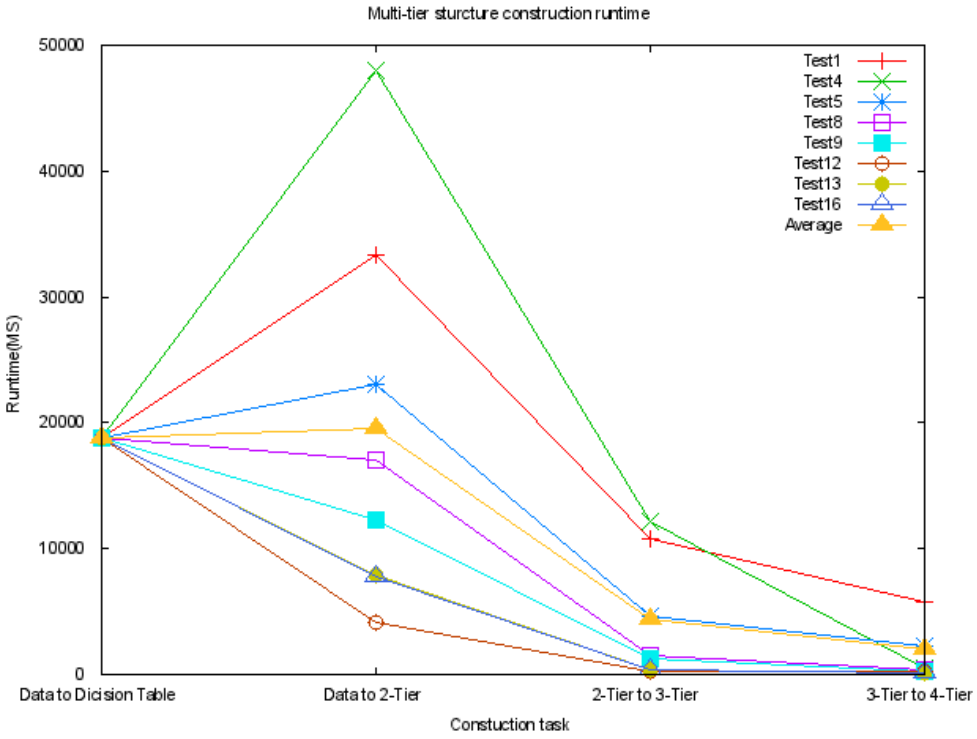


Figure 6.10: Construction time of different multi-tier structure

6.5.1 Details of the network traffic data

The network traffic data used in the tests were part of the MAWI data traces, which were downloaded from the MAWI website (*http://mawi.wide.ad.jp/mawi/samplepoint-B/20060303/*). Some examples of these data are shown in Table 6.15. Each of these records consists of a source IP address, a target IP address and a support of this traffic record. Both the source and target IP address were in one column in the raw data. These IP addresses were divided into single attributes according to their IP address domain. For example, the source IP address 196.244.108.25 was divided into four attributes: 196, 244, 108 and 25.

Due to this configuration, it is difficult for traditional pattern mining approaches–

Source IP	Target IP	Support
6.196.217.210	140.223.50.64	27536
196.44.176.223	165.28.231.200	23677
...	...	...
196.244.108.25	215.43.148.136	1072
200.208.178.1	148.179.135.12	1055

Table 6.15: Network traffic data

such as the Apriori algorithm–to perform efficiently in the scenario of network traffic analysis. This is because there are too many items in network traffic data and there is a specific requirement to define useful patterns in terms of network traffic analysis. For each attribute of the IP address, there are 256 possible values and, using the traditional pattern-based approach, each of these values of each attribute must be treated as an individual item for network traffic analysis. Thus, there would be a maximum of $256 \times 8 = 2{,}048$ items in the data, which would make the performance of the traditional pattern mining methods inefficient.

Moreover, there are some special requirements for the desired patterns in network traffic analysis. For example, only patterns containing both the attributes from the source IP address and destination IP address are useful, while patterns that only include the source IP address or destination IP address are useless. Thus, using the traditional pattern mining technology–such as the Apriori algorithm–will generate some useless patterns and thus waste computational resources. Therefore, pattern mining-based approaches cannot perform on the network traffic data directly, and were not applied in this set of experiments.

The granule mining approach is clearly suitable for the scenario of network traffic analysis. The source IP address and destination IP address can be easily

transformed into the condition and decision attributes. Each value in the IP address is treated as one of the attributes' defining granules, such that the multi-tier structure can be constructed by dividing the IP address into individual domains.

6.5.2 Details of the experiments

For the scalability experiment, the network traffic data provided a larger data volume than did Foodmart data set. The total number of transactions used in this experiment was 250,000. Several tests were performed on different numbers of transactions, from as small as 5,000, to the relatively large number of 250,000. The results reflect the trends of the performance changes, as well as the increment of the number of test data, such as changes in the number of granules and meaningless rules as the number of transactions increased.

In order to demonstrate the effect of the meaningless rules in a more understandable manner, it is preferable to use data that can easily show how meaningless rules help improve performance. The properties of network traffic data enable these data to satisfy this requirement because the relationship between the attributes of the IP address are obvious; thus, the effect of the meaningless rule can be illustrated explicitly and clearly.

In the experiment, a two-tier structure was first built using the source IP address as condition attribute and the target IP address as decision attribute. The condition attributes were then divided into smaller groups to define smaller granules, and the multi-tier structure was built according to the divisions.

According to the property of the IP address, each attribute represents a domain or region of the address. Thus, the granules of each tier represent the IP addresses of certain domains or regions. It should be noted that, different to the Foodmart 2005 data set, each attribute had multiple values from one to 255. In particular, the first three domains of the IP address were used to define the C_i granules, while the last domains were used to define the C_j granules. Further, the first two domains and third domain were used to generate the $C_{i,1}$ and $C_{i,2}$ granules, respectively.

For example, for the first transaction shown in Table 6.15, the whole source IP address (6.196.217.210) was used as the condition attribute to generate the C-granule. The address was then divided into two parts (6.196.217 and 210) to create the C_i and C_j granules. Finally, the sub-address (6.196.217) was further divided into two parts (6.196 and 217) to produce the $C_{i,1}$ and $C_{i,2}$ granules.

6.5.3 Results

Table 6.16 shows the numbers of granules in each tier in the multi-tier structure. As with the experiment on the Foodmart 2005 data, the number of granules decreased when more tiers were constructed. The granules were greatly reduced even with only one additional tier. However, it should be noted that, in contrast to the Foodmart 2005 data set, the number of granules for one tier was less or equal to 256 when the tier consisted of only one attribute. This was because there were 256 values of one attribute of one single IP address domain.

	Data	All	C	D	C_i	C_j	$C_{i,1}$	$C_{i,2}$
Test1	250000	92482	24887	45551	16363	256	4271	256
Test2	200000	64467	19546	27906	11970	256	3656	256
Test3	100000	40772	13952	19234	8222	256	2993	256
Test4	50000	21346	8674	10594	4967	256	2087	256
Test5	25000	10658	4938	5755	2968	256	1435	256
Test6	10000	6012	3049	3409	1997	256	1081	256
Test7	5000	3290	1701	1995	1249	254	788	256

Table 6.16: Granule number in tiers for network traffic data

Table 6.16 shows the results from the test on different numbers of the total data, from 1,000 data rows to 250,000 data rows. The results reflect the trends of the number of granules when the number of data increased. The number of granules did not increase greatly, while the number of transactions became much larger. For example, when the number of data increased from 100,000 to 200,000, the number of granules did not increase by double. The number of granules increased by 8,672 for D tier and 3,255 for C_i tier. In particular, for the tier that consisted of only one attribute, the number of granules remained at a number less than 256, regardless of the changes in the total data, because this number was the maximum number of different granules that could be generated for this configuration.

Table 6.17 presents the number of meaningless rules filtered in multi-tier structures with different numbers of tiers in the structure. The tests were also performed on different numbers of data rows, from 1,000 to 250,000. The results reflect the trend that, with more tiers in the structure, more meaningless rules could be filtered. Further, more general rules could be obtained if there were more tiers in the multi-tier structure. With more general rules, more meaningless

Criteria	$C_i$to D		$C_{i,1}$to D	
Rule	Meaningful	Meaningless	Meaningful	Meaningless
Test1	88931	2776	90053	1654
Test2	62249	971	63134	1333
Test3	39801	1216	40213	559
Test4	20890	456	21086	260
Test5	10452	206	10548	110
Test6	5910	102	5950	62
Test7	3265	25	3272	18

Criteria	$C_i$to D and $C_{i,1}$to D	
Rule	Meaningful	Meaningless
Test1	88527	3180
Test2	61897	2570
Test3	39630	1142
Test4	20807	539
Test5	10423	235
Test6	5895	117
Test7	3262	28

Table 6.17: Meaningful and meaningless rule numbers for network traffic data

rules could be pruned. Finally, comparing the results obtained through the tests on different numbers of transactions showed that the number of meaningless rules pruned increased with the increase in the number of transactions.

General rule	Support	Meaningless rule	Support
215.35.250 → 281.92.187.234	0.33293	215.35.250.88 → 218.92.187.234	0.30094
5.210.14 → 140.223.139.34	0.13723	5.210.14.140 → 140.223.139.34	0.095576
204.91.186 → 215.35.250.88	0.66829	204.91.186.178 → 215.35.250.88	0.45077

Table 6.18: Examples of meaningless rules and their general rule

Table 6.18 lists three examples of meaningless rules found in the test, with

their corresponding general rules. For example, the rule (204.91.186.178 → 215.35.250.88) is a meaningless rule, while the rule (204.91.186 → 215.35.250.88) is its general rule. According to the definition of meaningless rules, this rule's confidence was lower than the confidence of its general rule. In this example, the confidence of the meaningless rule was 0.45077, while its general rule had a greater confidence of 0.66829. Thus, this meaningless rule removal had the following implications. The rule (204.91.186.178 → 215.35.250.88) with a confidence of 0.45077 indicates that 45% of the visitors from the IP address 204.91.186.178 visited the target address. However, the general rule (204.91.186 → 215.35.250.88) with a confidence of 0.66829 indicates that 65% of the visitors from the domain of 204.91.186–which includes the visitors from the address 204.91.186.178–visited the same target. That is, the portion of visitors from the address 204.91.186.178 that visited the target address was not as significant as the portion of visitors from the larger domain visiting the same target. Therefore, although the rule (204.91.186.178 → 215.35.250.88) could be strong, it is not meaningful under such circumstances and should not be included in the results presented to the user.

6.6 Discussion

The experiments conducted have demonstrated the performance of the granule mining approach from several aspects, including the space complexity, time complexity, restoration error rate for support estimation, and scalability. This section

discusses the findings from the results of the experiments regarding these aspects.

For space complexity, the first discovery was that the number of granules in one tier was less when the number of attributes of the granule in that tier were less. The results shown in Tables 6.7 and 6.8 support this finding. For example, in Test 1, the C tier had 22 attributes, and there were 7,521 granules in this tier. C_i tier had smaller granule and attribute numbers of 6,222 and 21. When the number of attributes for $C_{i,1}$ tier reduced to 20, the number of granules reduced to 4,506. The total granule numbers in these two examples were 6,224 and 4,510, respectively. Another observation to support this finding was the number of granules in the D tier throughout all 16 tests. When the numbers of attributes were set to one, two and four, the numbers of granules in D tier were two, four and 16, respectively. Finally, as the attributes of D-granule increased to 11, 17 and 20, the numbers of granules also increased to 424, 2,933 and 5,622. All the results of the 16 tests showed the same trends as these two examples.

The second finding involved the time complexity related to the multi-tier structure construction. The results of the runtime led to two conclusions about the performance of the granule mining approach. The first was that, when there are more tiers in the structure, less time is consumed to generate the new tiers. In the results, the time used to create the three-tier structure from the two-tier structure was 2,593 ms, while it only took 171 ms to build the four-tier structure from the three-tier structure. Another finding was that the time consumed by the granule approach was much less than that of the pattern mining approach.

Therefore, we conclude that pattern-based post-processing approaches, such as pattern summarisation, consume more time than the granule approach.

The third discovery, from the experiment on restoration error rate, was that the granule mining approach is an efficient method of calculating the estimated support. First, from the experiment results for the runtime tests, the granule approach consumes much less time to construct the multi-tier structure than the time used to obtain frequent and closed patterns for the post-process approaches. Second, the results of the restoration error rate experiment reflected that the performance of the granule approach in calculating the estimated support is better than pattern summarisation when using a small number of profiles, such as 200, 500 and 750. It should be noted that, by using decision tables or multi-tier structures to calculate the estimated support, it is possible to achieve a zero restoration error rate.

The final point is about the scalability of the granule mining approach. From the experiments on the network traffic data, we concluded that the granule mining and multi-tier structure have good scalability. The experiment conducted on different sizes of data demonstrated that the number of granules did not change dramatically with the increase in transactions. Moreover, the number of granules changed even less if there were more tiers in the structure. Finally, when there were more transactions in the data, more meaningless rules could be filtered, and the result set can remain a reasonable size.

However, there are some limitations of this granule-based model. First, we on-

ly discussed the relationship between granules and closed patters in this method. Thus, we did not investigate the possibility of using a non-closed pattern. Regarding non-closed patterns, there is a concept called a 'generator' that uses non-closed patterns to deduce closed patterns and condensed or reliable representation rules [73, 111]. A generator of a closed pattern is the minimal pattern that its closure equals to the closed pattern. Using generators, the number of patterns can be reduced because only minimal patterns are included in the results. However, generators are not compressed representations of patterns, and the relationship between generators and granules was not investigated in this research. Therefore, we did not employ generators as a subject to compare with in our evaluation.

Another limitation of the proposed granule-based method is related to rule coverage. For the granule-based model, the antecedent and consequence of association is confined to the definition of condition and decision granules. That is, the condition of a rule must derive from condition granules, and the consequence of a rule must derive from decision granules. For the pattern-based method, there is no such limitation–rules are generated from patterns with flexible subset combinations. This is an obvious disadvantage of the granule-based method. This disadvantage also can lead to some false negative issues such that some interesting rules–where the antecedent contains attributes from the decision granules–are possibly missed. In order to obtain rules using attributes defining current decision granules as antecedent attribute, the granules and multi-tier structure must

be constructed with modified definitions accordingly.

Chapter 7

Conclusions and Future Work

7.1 Conclusion

Knowledge discovery in databases (KDD)–or data mining–is believed to be a promising technology to solve the knowledge acquisition bottleneck problem. However, guaranteeing the quality of the discovered knowledge is a challenging task. The essential research issue is how to represent meaningful association rules efficiently. In terms of pattern mining, this issue can be described as the problem of the interpretability of the discovered patterns and the redundancy of association rules. The obstacles for the application of association rule mining include the overwhelmingly large volume of discovered patterns and rules, the lack of structural information along the mining process, and the incompleteness

of the knowledge coverage.

This research used granule mining to solve these challenging problems. Granules were used instead of patterns in the knowledge discovery process. With the definitions of granules, attributes from multiple dimensions were divided into tiers so that the large multi-dimensional database could be compressed into granules at each tier. A series of theories and theorems was also defined to formally interpret granules in terms of patterns, and describe the derivation of patterns from granules.

Moreover, multi-tier structures were provided to efficiently organise granules with different sizes, and association mappings were generated to illustrate the complicated correlation between different data tiers (granules), where the antecedent and consequent of an association rule are both granules. The concepts of general rules and meaningless rules were also presented with the multi-tier structures.

For the association mappings, we provided the formal concepts of basic association mapping and derived mappings, including the definition of deriving these derived mappings from basic mappings. We also presented a formal description of the relationship between granules and transactions for basic mappings, as well as the relationship between the basic mappings and derived mappings. Based on this relationship, methods were developed to calculate the derived mappings from the basic mappings.

In this research, we also provided a method to estimate patterns' support

based on granules and mappings. According to different cases of how the input pattern was derived from the granules, several corresponding calculations were designed to estimate the support. Moreover, we proved that using granules and mappings to calculate the estimate support can achieve a zero error rate.

In order to implement the multi-tier structure, a set of algorithms were developed to perform the construction. These algorithms included the generation of basic and derived association mappings, the retrieval of decision rules from the multi-tier structure and the filtering of meaningless rules. Moreover, analysis was undertaken to demonstrate the efficiency of the algorithms. For support estimation, each case has a corresponding algorithm to perform the calculation. Specifically, there are different algorithms designed for two-tier structures and for structures with three or more tiers.

With the algorithms, a set of experiments were conducted against different measures to evaluate the performance of the proposed approach. One of the test data was the Foodmart 2005 data collection, which are multi-dimensional retail sales data. The data of the sales cube were transformed into the information table from which the multi-tier structures were built. The examination measure included time and space complexity for comparison with pattern-based methods. Moreover, several tests were conducted to indicate the number of meaningless rules filtered under different configurations. For support estimation, the estimation error rate was used as the performance measure. The test calculated the estimated support from the structures with different tiers, and compared the

performance with the pattern summarisation method with different numbers of profiles. Another set of experiments was performed on the data of network traffic records. This set of experiments proved the performance of the granule mining approach in regard to scalability. The effect of the meaningless rule was also illustrated using some examples generated from this data set.

This study has proved that the proposed multi-tier structures for describing associations between granules are efficient. The experimental results show that the multi-tier structures can use a very small space to store possible associations, and can be created efficiently. Another exciting outcome is that the proposed multi-tier structure enables users to discuss association rules and their general rules. In this way, meaningless association rules can be justified according to the relationship between association rules and their general rules. For the estimated support, compared with the pattern summarisation approach, the proposed multi-tier granule mining achieves the best performance with a zero restoration error rate. In conclusion, this research provides significant contributions for the interpretation of discovered knowledge in databases.

7.2 Future work

There are several aspects that could be further explored to continue improving the performance of the multi-tier granule mining approach. First, the derived association mappings discussed in the current research only occurred in the condition granules, and it is possible to discuss the derived association mapping for the

decision granules. Further, the relationship of the derived association mappings for both the condition and decision granules needs to be considered. Second, although the current approach used data schema and dimensional hierarchy information to define the granules and association mappings so that more semantic meanings could be interpreted, there is scope to develop a mechanism to better present this information to the end user. Finally, according to the experiment on the network traffic data, there are special cases in which the granule mining approach is more suitable than the traditional pattern-based method. Thus, in future, it would be worth researching further real scenarios to which granule mining could be applied.

Bibliography

[1] F. Afrati, A. Gionis, and H. Mannila. Approximating a collection of frequent sets. In *Proceedings of KDD'04*, pages 12–19, Seattle, Washington, USA, 2004.

[2] R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In *Proceedings of SIGMOD'93*, pages 207–216, Washington, DC, USA, 1993.

[3] R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, and A. I. Verkamo. Fast discovery of association rules. In U. M. Fayyad, G. Piatetsky, P. Smyth, and R. Uthurusamy, editors, *Advance in Knowledge Discovery and Data Mining*, pages 307–328. AAAI/MIT Press, 1996.

[4] R. Agrawal and R. Srikant. Fast algorithms for mining association rules in large databases. In *Preceedings of VLDB'94*, IBM Almaden Research Center, San Jose, CA, USA, 1994.

[5] K. M. Ahmed, N. M. El-Makky, and Y. Taha. Effective data mining: a data warehouse-backboned architecture. In *Proceedings of the 1998 conference*

of the Centre for Advanced Studies on Collaborative research, pages 1 – 11, Toronto, 1998.

[6] A. Amir, Y. Aumann, R. Feldman, and M. Fresko. Maximal association rules: A tool for mining associations in text. *Journal of Intelligent Informaiton Systems*, 25(3):333 – 345, 2005.

[7] J. Bayardo. Efficiently mining long patterns from databases. In *Proceedings of SIGMOD'98*, pages 85–93, 1998.

[8] R. Bayardo, R. Agrawal, and D. Gunopulos. Constraint-based rule mining in large, dense databases. *Data Mining and Knowledge Discovery*, 4:217–240, 2000.

[9] J. G. Bazan, S. H. Nguyen, H. S. Nguyen, and A. Skowron. Rough set methods in approximation of hierarchical concepts. In *Proceedings of 4th International Conference on Rough Sets and Current Trends in Computing*, pages 346–355. Springer-Verlag, 2004.

[10] C. Bucila, J. Gehrke, D. Kifer, and W. White. Dualminer: a dual-pruning algorithm for itemsets with constraints. In *Proceedings of KDD'02*, pages 42–51, Alberta, Canada, 2002.

[11] T. Calders and B. Goethals. Mining all non-derivable frequent itemsets. In *Proceedings of 2002 European Conference on Principles of Data Mining and Knowledge Discovery*, pages 74–85, 2002.

[12] U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthrusamy. *Advances in knowledge discovery and data mining.* Menlo Park, California, AAAI Press/ The MIT Press, 1996.

[13] R. Feldman, Y. Aumann, A. Amir, A. Zilberstein, and W. Kloesgen. Maximai association ruies: a new tooi for mining for keyword cooccurrences in document collections. In *Proceedings of KDD'97*, pages 167–170, 1997.

[14] B. Ganter and R. Wille. *Formal Concept Analysis: Mathematical Foundation.* Springer-Verlag, 1999.

[15] E. Ge, R. Nayak, Y. Xu, and Y. Li. Data mining for lifetime prediction of metallic components. In *Proceedings of Australasian Data Mining Conference 2006*, pages 75–82, Sydney, Australia, 2006.

[16] P. Giuseppe and L. L. Pier. Hierarchy-based mining of association rules in data warehouses. In *Proceedings of ACM symposium on Applied computing*, pages 307 – 312, Como, 2000.

[17] B. Goethals and H. T. J Muhonen. Mining non-derivable association rules. In *Proceedings of SDM'05*, pages 239–249, 2005.

[18] S. Goil and A. Choudhary. A parallel scalable infrastructure for olap and data mining. In *Proceedings of IDEAS'99*, pages 178 – 186, Montreal, 1999.

[19] J. Guan, D. Bell, and D. Liu. The rough set approach to association rules. In *Proceedings of 3rd IEEE International Conference on Data Mining (ICDM)*, pages 529–532, Melbourne, Florida, USA, 2003.

[20] J. W. Guan and D. A. Bell. Rough computational methods for information systems. *Artificial Intelligence*, 105:77–103, 1998.

[21] R. Gupta, G. Fang, and B. Field. Quantitative evaluation of approximate frequent pattern mining algorithms. In *Proceedings of KDD08*, pages 301–309, Las Vegas, Nevada, USA, 2008.

[22] J. Han, H. Cheng, D. Xin, and X. Yan. Frequent pattern mining: current status and future directions. *Data Mining and Knowledge Discovery*, 15(1):55–86, 2007.

[23] J. Han and Y. Fu. Discovery of multiple-level association rules from large databases. In *Proceedings of VLDB'95*, pages 420–431, Zurich, Switzerland, 1995.

[24] J. Han and Y. Fu. Mining multiple-level association rules in large databases. *IEEE ransaction on Knowledge and Data Engineering*, 11(5):798–805, 1999.

[25] J. Han and M. Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers, 2nd edition, 2006.

[26] J. Han, L. V. Lakshmanan, and R. T. Ng. Constraint-based multidimensional data mining. *IEEE Computer*, 32(8):46–50, 1999.

[27] J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. In *Proceedings of SIGMOD'00*, pages 1–12, Dallas, TX, USA, 2000.

[28] J. Hendler and E. A. Feigenbaum. Knowledge is power: the semantic web vision. *Web Intelligence: research and development, Lecture Notes in Computer Science*, 2198:18–29, 2001.

[29] W. H. Inmon. The data warehouse and data mining. *Communications of the ACM*, 39:49–50, 1996.

[30] W. H. Inmon. *Building the Data Warehouse*. Wiley Technology Publishing, 2005.

[31] B. C. Jamie MacLennan, ZhaoHui Tang. *Data Mining with SQL Server 2008*. Wiley Publishing, Inc., 2009.

[32] J. Janas. An enhanced a priori algorithm for mining multidimensional association rules. In *Proceedings of The 25th International Conference on Information Technology Interfaces*, pages 193 – 198, Carvat, 2003.

[33] R. Jin, M. Abu-Ata, Y. Xiang, and N. Ruan. Effective and efficient itemset pattern summarization: Regression-based approaches. In *Proceedings of KDD08*, pages 399–407, Las Vegas, Nevada, USA, 2008.

[34] R. Jin, Y. Xiang, and L. Liu. Cartesian contour: a concise representation for a collection of frequent sets. In *Proceedings of KDD'09*, pages 417–426, 2009.

[35] M. Kamber, J. Han, and Y. J. Chiang. Metarule-guided mining of multi-dimensional association rules using data cubes. In *Proceedings of KDD'97*, pages 207–210, Newport Beach, 1997.

[36] Y. Ke, J. Cheng, and W. Ng. Mining quantitative correlated patterns using an information-theoretic approach. In *Proceedings of KDD'06*, pages 227–236, Philadelphia, Pennsylvania, USA, 2006.

[37] R. Kimball and M. Ross. *The Data Warehouse Toolkit - The Complete Guide to Dimensional Modelling*. John Wiley & Sons, New York, 2 edition, 2002.

[38] M. Kryszkiewicz, H. Rybinski, and M. Gajek. Dataless transitions between concise representations of frequent patterns. *Journal of Intelligent Information Systems*, 22(1):41–70, 2004.

[39] A. J. T. Lee, W. Lin, and C. Wang. Mining association rules with multi-dimensional constraints. *Journal of Systems and Software*, 79(1):79–92, 2006.

[40] B. Lent, A. N. Swami, and J. Widom. Clustering association rules. In *Proceedings of ICDE'97*, pages 220–231, Washington, 1997.

[41] C. K.-S. Leung, L. V. S. Lakshmanan, and R. T. Ng. Exploiting succinct constraints using fp-trees. *SIGKDD Explorations*, 4(1):40–49, 2002.

[42] Y. Leung, M. M. Fischer, W. Wu, and J. Mi. A rough set approach for the discovery of classification rules in interval-valued information systems. *International Journal of Approximate Reasoning*, 47:233–246, 2008.

[43] T. Li and D. Ruan. An extended process model of knowledge discovery in database. *Journal of Enterprise Information Management*, 20(2):169–177, 2007.

[44] Y. Li. Extended random sets for knowledge discovery in information system. In *Proceedings of the 9th International Conference on Rough Sets, Fuzzy Sets, Data Miing and Granular Computing*, pages 524–532, Chongqing, China, 2003.

[45] Y. Li, A. Algarni, and N. Zhong. Mining positive and negative patterns for relevance feature discovery. In *Proceeding of KDD'10*, pages 753–762, 2010.

[46] Y. Li, S. C. K. Shiu, S. K. Pal, and J. N. K. Liu. A rough set-based case-based reasoner for text categorization. *International Journal of Approximate Reasoning*, 41:229–255, 2006.

[47] Y. Li, S.-T. Wu, and Y. Xu. Deploying association rules on hypothesis spaces. In *Proceedings of CIMCA'04*, pages 769–778, 2004.

[48] Y. Li, W. Yang, and Y. Xu. Multi-tier granule mining for representations of multidimensional association rules. In *Proceedings of 6th IEEE International Conference on Data Mining (ICDM)*, pages 953–958, Hong Kong, 2006.

[49] Y. Li and N. Zhong. Interpretations of association rules by granular computing. In *Proceedings of 3rd IEEE International Conference on Data Mining (ICDM)*, pages 593–596, Melbourne, Florida, USA, 2003.

[50] Y. Li and N. Zhong. Mining ontology for automatically acquiring web user information needs. *IEEE Transactions on Knowledge and Data Engineering*, 18(4):554–568, 2006.

[51] Y. Li and N. Zhong. Mining rough association from text documents. In *Proceedings of 5th International Conference on Rough Sets and Current Trends in Computing*, pages 368–377, Kobe, Japan, 2006.

[52] Y. Li and N. Zhong. Mining rough association from text documents for web information gathering. *Springer Transactions on Rough Sets*, VII:103–119, 2006.

[53] Y. Li and N. Zhong. Rough association rule mining in text documents for acquiring web user information needs. In *Proceedings of IEEE/WIC/ACM International Conference on Web Intelligence 2006*, Hong Kong, 2006.

[54] Y. Li, X. Zhou, P. Bruza, Y. Xu, and R. Y. Lau. A two-stage text mining model for information filtering. In *Proceeding of 17th ACM Conference on*

Information and Knowledge Management (CIKM), pages 1023–1032, New York, NY, USA, 2008.

[55] P. Lingras and Y. Y. Yao. Data mining using extensions of the rough set model. *Journal of the American Society for Information Science*, 49:415–422, 1998.

[56] P. Lingras and Y. Y. Yao. Time complexity of rough clustering: Gas versus k-means. In *Proceedings of 3rd International Conference on Rough Sets and Current Trends in Computing*, pages 263–270, Malvern, PA, USA, 2002.

[57] B. Liu, Y. Li, and Y.-C. Tian. Discovering novel knowledge using granule mining. In *Proceedings of 8th International Conference on Rough Sets and Current Trends in Computing*, pages 380–387, 2012.

[58] B. Liu, Y. Li, and K. Wang. Granule mining and its application for network traffic characterization. *Rough Sets and Current Trends in Computing, Lecture Notes in Computer Science*, 16:333–343, 2012.

[59] B. Liu, K. Zhao, J. Benkler, and W. Xiao. Rule interestingness analysis using olap operations. In *Proceedings of KDD'06*, pages 297–306, Philadelphia, Pennsylvania, USA, 2006.

[60] G. Liu, H. Zhang, and L. Wong. Finding minimum representative pattern sets. In *Proceedings of KDD'12*, 2012.

[61] J. Liu, S. Paulsen, X. Sun, and W. Wang. Mining approximate frequent itemsets in the presence of noise: algorithm and analysis. In *Proceedings of SIAM International Conference on Data Mining*, 2006.

[62] H. Lu, J. Han, and L. Feng. Stock movement prediction and n-dimensional inter-transaction association rule. In *Proceedings of DMKD'98*, volume 12, pages 1–7, Seattle, 1998.

[63] H. Mannila, H. Toivonen, and A. I. Verkamo. Efficient algorithms for discovering association rules. In *Proceedings of AAAI'94 Workshop on Knowledge Discovery in Databases*, pages 181–192, Seattle, WA, USA, 1994.

[64] Q. Mei, D. Xin, H. Cheng, J. Han, and C. Zhai. Generating semantic annotations for frequent patterns with context analysis. In *Proceedings of KDD'06*, pages 337–346, Philadelphia, Pennsylvania, USA, 2006.

[65] R. B. Messaoud, S. L. Rabaseda, O. Boussaid, and R. Missaoui. Enhanced mining of association rules from data cubes. In *Proceedings of 9th ACM international workshop on Data warehousing and OLAP*, pages 11–18, Arlington, Virginia, USA, 2006.

[66] A. Michail. Data mining library reuse patterns using generalized association rules. In *Proceedings of the 22nd international conference on Software engineering*, pages 167–176, Limerick, 2000.

[67] R. S. Monteiro, G. Zimbro, H. Schwarz, B. Mitschang, and J. M. Souza. Building the data warehouse of frequent itemsets in the dwfist approach. *Foundations of Intelligent Systems*, 3488:294–303, 2005.

[68] M. Moshkov, A. Skowron, and Z. Suraj. Extracting relevant information about reduct sets from data tables. *Transactions on Rough Sets IX*, LNCS 5390:200–211, 2008.

[69] S. Nestorov and N. Jukic. Ad-hoc association-rule mining within the data warehouse. In *Proceedings of the 36th Annual Hawaii International Conference on System Sciences*, pages 10 – 19, Hawaii, 2003.

[70] R. T. Ng, L. V. S. Lakshmanan, J. Han, and A. Pang. Discovery of multiple-level association rules from large databases. In *Proceedings of SIGMOD'98*, pages 13–24, Seattle, Washington, USA, 1998.

[71] J. S. Park, M. S. Chen, and P. S. Yu. An effective hash-based algorithm for mining association rules. In *Proceedings of SIGMOD'95*, pages 175–186, San Jose, CA, USA, 1995.

[72] N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Discovering frequent closed itemsets for association rules. In *Proceedings of the 7th International Conference on Database Theory*, pages 398–416, 1999.

[73] N. Pasquier, R. Taouil, Y. Bastide, G. Stumme, and L. Lakhal. Generating a condensed representation for association rules. *Journal of Intelligent Informaiton Systems*, 24(1):29–60, 2005.

[74] Z. Pawlak, , and A. Skowron. Rough sets and boolean reasoning. *Information Sciences*, 177:41–73, 2007.

[75] Z. Pawlak. Rough sets. *International Journal of Computer Information and Sciences*, 11:341–356, 1982.

[76] Z. Pawlak. In pursuit of patterns in data reasoning from data - the rough set way. In *Proceedings of 3rd International Conference on Rough Sets and Current Trends in Computing*, pages 1–9, Malvern, PA, USA, 2002.

[77] Z. Pawlak. Rough sets and intelligent data analysis. *International Journal of Computer Information and Sciences*, 147:1–12, 2002.

[78] Z. Pawlak. Decision trees and flow graphs. In *Proceedings of 5th International Conference on Rough Sets and Current Trends in Computing*, pages 1–11, Kobe, Japan, 2006.

[79] Z. Pawlak, J. Grzymala-Busse, R. Slowinski, and W. Ziarko. Rough sets. *Communications of The ACM*, 38(11):89–95, 1995.

[80] J. Pei, G. Dong, W. Zou, and J. Han. On computing condensed frequent pattern bases. In *Proceedings of 2nd IEEE International Conference on Data Mining (ICDM)*, pages 378–385, 2002.

[81] J. Pei and J. Han. Constrained frequent pattern mining: a pattern-growth view. *SIGKDD Explorations*, 4(1):31–39, 2002.

[82] J. Pei, J. Han, and L. V. S. Lakshmanan. Pushing convertible constraints in frequent itemset mining. *Data Mining and Knowledge Discovery*, 8(3):227–252, 2004.

[83] C. Perng, H. Wang, S. Ma, and J. L. Hellerstein. Discovery in multi-attribute data with user-defined constraint. *SIGKDD Explorations*, 4(1):56–64, 2002.

[84] A. K. Poernomo and V. Gopalkrishnan. Cp-summary: a concise representation for browsing frequent itemsets. In *Proceedings of KDD'09*, pages 687–696, 2009.

[85] S. Raja and S. Vijayalakshmi. Multidimensional frequent pattern mining using association rule based constraints. *Lecture Notes in Computer Science*, 3816:585 – 591, 2005.

[86] S. Ruggieri. Frequent regular itemset mining. In *Proceedings of KDD'10*, pages 263–272, 2010.

[87] A. Skowron. Rough sets and vague concepts. *Fundamenta Informaticae*, 64(1-4):417–431, 2005.

[88] A. Skowron. Approximate reasoning in mas: rough set approach extended abstract (keynote talk). In *Proceedings of IEEE/WIC/ACM International Conference on Web Intelligence 2006*, 2006.

[89] A. Skowron and J. Grzymala-Busse. From rough set theory to evidence theory. In R. R. Yaeger, M.Fedrizzi, and J. Kacprzyk, editors, *Advances in the Dempster-Shafer theory of evidence*, pages 193–236. Wiley, New York, 1994.

[90] A. Skowron, H. Wang, A. Wojna, and J. G. Bazan. A hierarchical approach to multimodal classification. In *Proceedings of the 10th International Conference on Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing (RSFDGrC'05)*, pages 119–127, 2005.

[91] R. Srikant and R. Agrawal. Mining quantitative association rules in large relational tables. In *Proceedings of SIGMOD'96*, pages 1–12, Montreal, 1996.

[92] P. N. Tan and V. Kumar. Discovery of indirect associations from web usage data. In N. Zhong, J. Liu, and Y. Y. Yao, editors, *Web Intelligence*, pages 128–152. Springer-Verlag, 2003.

[93] E. Tsang and Z. Suyun. Decision table reduction in kdd: fuzzy rough based approach. *Transactions on Rough Sets XI*, LNCS 5946:177–188, 2010.

[94] S. Tsumoto and S. Hirano. Visualization of rule's similarity using multidimensional scaling. In *Proceedings of 3rd IEEE International Conference on Data Mining (ICDM)*, pages 339–346, Melbourne, Florida, USA, 2003.

[95] A. K. H. Tung, H. Lu, J. Han, and L. Feng. Efficient mining of inter-transaction association rules. *IEEE Transactions on Knowledge and Data Engineering*, 15(1):43–56, 2003.

[96] P. Tzvetkov, X. Yan, and J. Han. Tsp: Mining top-k closed sequential patterns. In *Proceedings of 3rd IEEE International Conference on Data Mining (ICDM)*, pages 347–354, Melbourne, Florida, USA, 2003.

[97] F. Verhein and S. Chawla. Geometrically inspired itemset mining. In *Proceedings of 6th IEEE International Conference on Data Mining (ICDM)*, pages 655–666, Hong Kong, 2006.

[98] C. Wang and S. Parthasarathy. Summarizing itemset patterns using probabilistic models. In *Proceedings of KDD'06*, pages 730–735, Philadelphia, Pennsylvania, USA, 2006.

[99] W. Wang, J. Yang, and P. Yu. Efficient mining of weighted association rules. In *Proceedings of KDD'97*, pages 270–274, Boston, 1997.

[100] G. Webb. Filtered-top-k association discovery. *WIREs Data Mining and Knowledge Discovery*, 1(3):183–192, 2011.

[101] G. I. Webb and S. Zhang. K-optimal rule discovery. *Data Mining and Knowledge Discovery*, 10:39–79, 2004.

[102] A. Wojna. Combination of metric-based and rule-based classification. In *Proceedings of the 10th International Conference on Rough Sets, Fuzzy Sets,*

Data Mining, and Granular Computing (RSFDGrC'05), pages 501–511, 2005.

[103] S.-T. Wu, Y. Li, and Y. Xu. Deploying approaches for pattern refinement in text mining. In *Proceedings of 6th IEEE International Conference on Data Mining (ICDM)*, pages 1157–1161, Hong Kong, 2006.

[104] S.-T. Wu, Y. Li, Y. Xu, B. Pham, and P. Chen. Automatic pattern-taxonomy extraction for web mining. In *Proceedings of IEEE/WIC/ACM International Conference on Web Intelligence 2004*, pages 242–248, Beijing, China, 2004.

[105] D. Xin, H. Cheng, X. He, and J. Han. Extracting redundancy-aware top-k patterns. In *Proceedings of KDD'06*, Philadelphia, Pennsylvania, USA, 2006.

[106] D. Xin, J. Han, X. Yan, and H. Cheng. Mining comressed frequent-pattern sets. In *Proceedings of VLDB'05*, pages 709 – 720, 2005.

[107] D. Xin, J. Han, X. Yan, and H. Cheng. On compressing frequent patterns. *Data and Knowledge Engineering*, 60:5–29, 2007.

[108] Y. Xu and Y. Li. Mining for useful association rules using the atms. In *Proceedings of CIMCA 2005*, pages 271 – 276, Vienna, 2005.

[109] Y. Xu and Y. Li. Generating concise association rules. In *Proceedings of 16th ACM Conference on Information and Knowledge Management (CIKM)*, pages 781–790, 2007.

[110] Y. Xu and Y. Li. Mining non-redundant association rule based on concise bases. *International Journal of Pattern Recognition and Artificial Intelligence*, 21:659–675, 2007.

[111] Y. Xu, Y. Li, and G. Shaw. Reliable representations for association rules. *Data and Knowledge Engineering*, 70(6):555–575, 2011.

[112] X. Yan, H. Cheng, J. Han, and D. Xin. Summarizing itemset patterns: A profilebased approach. In *Proceedings of KDD'05*, pages 314–323, Chicago, Illinois, USA, 2005.

[113] C. Yang, U. Fayyad, and P. S. Bradley. Efficient discovery of error-tolerant frequent itemsets in high dimensions. In *Proceedings of KDD'01*, pages 194–203, San Francisco CA USA, 2001.

[114] W. Yang, Y. Li, J. Wu, and Y. Xu. Granule mining oriented data warehousing model for representations of multidimensional association rules. *International Journal of Intelligent Information and Database Systems*, 2(1):125–145, 2008.

[115] Y.-P. O. Yang, H.-M. Shieh, G.-H. Tzeng, L. Yen, and C.-C. Chan. Combined rough sets with flow graph and formal concept analysis for busi-

ness aviation decision-making. *Journal of Intelligent Informaiton Systems*, 36:347–366, 2011.

[116] J. Yao and Y. Y. Yao. Induction of classification rules by granular computing. In *Proceedings of 3rd International Conference on Rough Sets and Current Trends in Computing*, pages 331–338. Springer-Verlag, 2002.

[117] Y. Y. Yao. A comparative study of formal concept analysis and rough set theory in data analysis. In *Proceedings of 4th International Conference on Rough Sets and Current Trends in Computing*, pages 59–68, 2004.

[118] Y. Y. Yao. Semantics of fuzzy sets in rough set theory. *Transactions on Rough Sets*, pages 297–318, 2004.

[119] Y. Y. Yao. Probabilistic rough set approximations. *International Journal of Approximate Reasoning*, 49(2):255–271, 2008.

[120] Y. Y. Yao, Y. Zhao, and R. B. Maguire. Explanation oriented association mining using rough set theory. In *Proceedings of the 8th International Conference on Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing (RSFDGrC'03)*, pages 165–172, 2003.

[121] J. Yuan, Y. Wu, and M. Yang. From frequent itemsets to semantically meaningful visual patterns. In *Proceedings of KDD'07*, pages 864–873, San Jose, California, USA, 2007.

[122] L. A. Zadeh. Granular computing - the concept of generalized constraint-based computing. In *Proceedings of 5th International Conference on Rough Sets and Current Trends in Computing*, pages 12–14, Kobe, Japan, 2006.

[123] M. J. Zaki. Mining non-redundant association rules. *Data Mining and Knowledge Discovery*, 9:223–248, 2004.

[124] S. Zhang, J. Zhang, X. Zhu, and Z. Huang. Identifying follow-correlation intemset-pairs. In *Proceedings of 6th IEEE International Conference on Data Mining (ICDM)*, pages 765–774, Hong Kong, 2006.

[125] Y. Zhao, Y. Y. Yao, and J. Yao. Level-wise construction of decison trees for classification. *International Journal of Software Engineering and Knowledge Engineering*, 16(1):103–126, 2006.